References to Existing CAN Literature

The following are some documents that give details of the CAN protocol, its applications and device and system implementations:

1. David J. Arnett, Intel Corporation, "A High Performance Solution for In-Vehicle Networking—Controller Area Network (CAN)", SAE 870823
2. Siegfried Dais and Michael Chapman, Robert Bosch GmbH, "Impact of Bit Representation on Transport Capacity and Clock Accuracy in Serial Data Streams", SAE International Congress and Exposition 1989, SAE 890532
3. I. Dudeck, U. Hipp and T. Raith, Daimler-Benz AG, "Investigations into the Electromagnetic Capability and Performance Evaluation of Databus Systems in Motor Vehicles", FISITA 1988
4. Uwe Kiencke, C. T. Cao and M. Litschel, Robert Bosch GmbH, "The Impact of Bit Representation on Noise Emissions in Automotive Networks", IEEE Conference on Automotive Electronics, Oct. 1987, London, Conference Publication No. 280
5. Uwe Kiencke and Siegfried Dais, Robert Bosch GmbH, "Application Specific Microcontroller for Multiplex Wiring", SAE 870515
6. Uwe Kiencke, Siegfried Dais and Martin Litschel, Robert Bosch, GmbH, "Automotive Serial Controller Area Network", SAE International Congress and Exposition 1986, SAE 860391
7. Uwe Kiencke, Siegfried Dais, Martin Litschel and Jan Unruh, Robert Bosch GmbH, "Error Handling Strategies for Automotive Networks", SAE 880587
8. Wolfhard E. Lawrenz, I+ME GmbH, "In-Vehicle Networking Requires New Development Tools", SAE International Congress and Exposition 1989, SAE 890533
9. Heikki Leppanen, Kone Corporation, "Performance Analysis of Non-Destructive Bitwise Arbitration Protocol"
10. Frederick H. Phail and David J. Arnett, Intel Corporation, "In-Vehicle Networking—Serial Communication Requirements and Directions", SAE International Congress and Exposition 1986, SAE 860390
11. Martin Zechnall and Gunther Kaiser, Robert Bosch GmbH, "A New MOTRONIC System with 16 Bit Micro Controller", SAE 891648
12. Robert Bosch GmbH, "Error Detection Capabilities of the CAN Protocol", ISO/TC22/SC3/WG1 document
13. Robert Bosch GmbH, "Specification of the CAN — Physical Layer for High-Speed-Application up to 1 Mbit/s" October 1989, presented to ISO on Oct 31 in Tokyo
14. Fiat Auto S.p.A., "EMC-Radiated Susceptibility: Bench Test on Two Electronic Units Linked with a CAN line using different kinds of Cables", ISO/TC22/SC3/WG1 document
15. Robert Bosch GmbH, "CAN Specification" version 1.0, 1987, ISO/TC22/SC3/WG1 document
16. Robert Bosch GmbH, "CAN Specification" version 1.1, 1989, ISO/TC22/SC3/WG1 document
17. Robert Bosch GmbH, "CAN Specification" version 1.2, 1990, ISO/TC22/SC3/WG1 document
18. Robert Bosch GmbH, "CAN Specification of the Data Link Layer and Physical Layer," June 1990, ISO/TC22/SC3/WG1 document

March 1991

82526 Serial Communications Controller Architectural Overview

Automotive

Order Number: 270678-002

82526 SERIAL COMMUNICATIONS CONTROLLER ARCHITECTURAL OVERVIEW

CONTENTS

PAGE

82526
SERIAL COMMUNICATIONS CONTROLLER ARCHITECTURAL OVERVIEW

Automotive

NOTE:
...ed version of the Architectural Overview will ...able in H2 '90.

...GENERAL FEATURES

- ...ltimaster Contention Based Bus Configuration
- ...s Access Priority by Message
- ...ect Oriented Communication
- ...2 Different Communication Objects
- ...w Latency Time for High Priority Messages
- ...werful Error Handling
- ...a Length 0–8 Bytes
- ...ssage Configuration Flexibility
- ...adcast Message Transfer
- ...on-Destructive Bitwise Arbitration
- ...wo 8-Bit Input/Output Ports
- ...RZ Coding/Decoding with Bit Stuffing
- ...ogrammable Transfer Rate up to 1MBit/Sec
- ...ogrammable Output Driver Configuration
- ...rogrammable Clock Output
- ...hree Additional $\overline{CS}$ Outputs
- ...eady Output
- ...lobal Interrupt Disable
- ...n-Board "Motel"
- ...4-Pin PLCC

...nctional Overview

...8526 Communication Controller is a highly inte-...ed VLSI device optimized for automotive in-vehicle ...orking requirements. The 82526 implements the ...N (Controller Area Network) protocol. CAN uses a ...timaster (contention based) bus configuration for ...nsfer of "communication objects" between the ...s of the network.

...ommunication object consists of an identifier along ...h control and data segments. The control segment ...ains all the information needed to transfer the mes-... The data segment contains from 0 to 8 bytes in a ...message. All Communication objects are stored ...ode's pseudo dual-port RAM (DPRAM). A ...smitting station broadcasts its message to all other ...ions on the network. After the message is received ...are checked, an acceptance filter at each station ...ides whether the message is intended for that node.

A message is accepted only if a communication object with the same identifier has been set up in the DPRAM for that station.

CAN not only manages the transmission and reception of messages but also the error handling, without any burden.

CAN features several error detection mechanisms. These include Cyclic Redundancy Check (CRC) and bit coding rules ("bit stuffing/destuffing"). The polynomial of the CRC has been optimized for automotive applications (short messages). If a message was corrupted by noise during transmission, it is not accepted at receiving stations. Current transmission status is monitored in the control segment of the appropriate communication object within the transmitting station, automatically initiating a repeated transmission in case of errors. CAN also has built-in mechanisms, called Fault Confinement, to locate error sources and to distinguish permanent hardware failures from occasional soft errors. Defective stations are switched off the bus, implementing a fail-safe behavior (thus, hardware errors will not let defective stations control the bus indefinitely).

The maximum length of data frames (including stuff bits) is limited to less than 127 bits. Thus the bus is free after a short latency time.

The CPU interfaces to the CAN through the 82526 pseudo dual port RAM (DPRAM). A priority access window is defined for each station by initializing the communication objects to be transmitted and received in its DPRAM. The CPU updates the information pertaining to a message by modifying the data segment of the proper communication object. In order to initiate a transfer, the transmission request bit has to be written into the respective control segment. The entire transmission procedure and eventual error handling is then done without any CPU involvement. Reception of messages is simple too. If a communication object has been set up for the message to be received, its data segment can be readily accessed by CPU read instructions. The control segment can be initialized such that the CPU is interrupted after every successful transfer.

The 82526 includes all the hardware modules to implement the CAN protocol on the bus (transfer and object layers). These hardware modules implement all the necessary features of a high performance serial communication protocol. When connected to a microprocessor, the 82526 performs the principal functions of the physical and data link layer. Figure 1 shows a block diagram of the 82526.

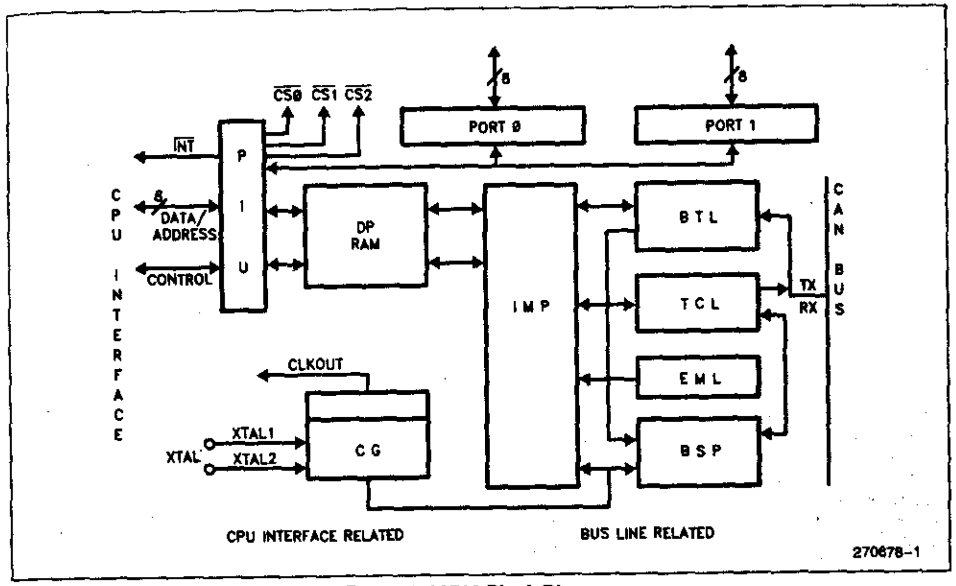


Figure 1. 82526 Block Diagram

1.2 82526/Host CPU Interface

The 82526 uses a multiplexed 8-bit address/data bus optimized for operating with Intel's microcontrollers and microprocessors. The 82526 has an on-board "Motel" circuitry which allows it to interface to 68xx architectures without any external logic.

As shown in Figure 1, the BSP, TCL, BTL and EML are related with Bus Line Logic. The IMP, RAM and PIU are related to the CPU Interface Logic. The logic blocks BSP, BTL, TCL and EML are referred to as the "Serial Interface Unit (SIU)".

The CPU communicates with the 82526 through the on-chip pseudo dual port RAM (DPRAM). Six RAM locations are reserved for global control and status registers. The on-chip pseudo dual port memory serves as the communication buffer interface between the CPU and the IMP. The CPU initializes the global status and control registers and creates a data structure (Communication Object(s)) within the communication buffer for reception and transmission of defined messages.

Commands, data and status exchanges between the CPU and the 82526 take place over the 8-bit parallel bus. The CPU writes data to the 82526 using $\overline{CS}$, ALE and $\overline{WR}$ signals and reads the 82526 received data and status information using the $\overline{CS}$, ALE and $\overline{RD}$ signals. The 82526 uses the interrupt line to alert the CPU of errors or successful transfers of data.

1.3 Interface Management Processor (IMP)

The IMP executes commands of the host controller and controls data transfers on the serial bus. Global status and control register bits, as well as the control bits of the communication objects are used primarily by the Interface Management Processor (IMP).

1.4 Bit Stream Processor (BSP)

The Bit Stream Processor controls the data stream between the IMP (parallel data) and the busline (serial data). The BSP also controls the Transceiver Control Logic (TCL) and the Error Management Logic (EML).

1.5 Bus Timing Logic (BTL)

This block monitors the busline through a differential input comparator and determines the bit timings related to the serial bus. The BTL synchronizes on a transition at the start of a frame and resynchronizes on further transitions during the reception of a frame. The BTL also provides programmable time segments to compensate for propagation delay times and phase shifts.

Transceiver Control Logic (TCL)

Transceiver Control Logic consists of bit stuffing, programmable output driver logic, CRC logic data shift registers. The BSP coordinates the indi-...al elements. Reception, arbitration, transmission ...rror signaling are performed by the TCL.

Error Management Logic (EML)

...rs are reported to the EML by the Bit Stream ...essor (BSP). The EML informs the BSP, TCL, and ... of error statistics.

Processor Interface Unit (PIU)

Processor Interface Unit is the interface of the ...6 to the host CPU. The 82526 can be interfaced to ... and other controllers without any additional logic ...ilizing the "Motel" circuit. The PIU consists of: ...iplexed 8-bit address/data bus, read/write con-... address latch enable (ALE), chip select ($\overline{CS}$), re-... $\overline{RST}$), interrupt output ($\overline{INT}$) and ready output ...Y). The PIU also consists of Port0/Port1 for addi-... 16 I/O port pins and chip select output signals ...-2) for additional system peripheral devices.

1.9 Clock Generator (CG)

The on-chip clock generator consists of an oscillator, clock divider register and driver circuit. The oscillator is a high-gain parallel resonance circuit with a frequency range from 1 MHz to 16 MHz. The oscillator can be driven by an external clock, crystal or a ceramic resonator depending on the application requirements. A host CPU can be driven from the clock output (CLKOUT). The clock output uses the programmable divider to select the output frequency.

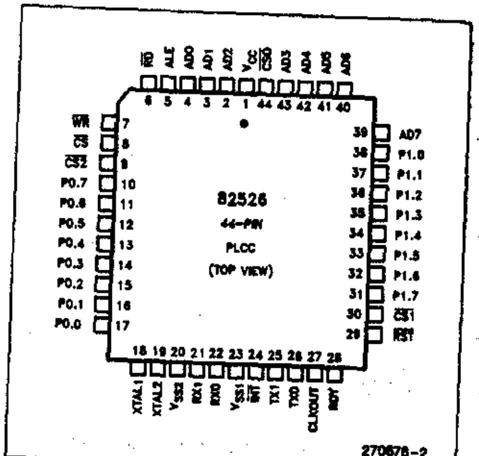


Figure 2. 44-PIN PLCC Package

PIN DESCRIPTION

Pin	Description
...C	Five volt logic supply (+5V).
...S1	Digital circuit ground (0V).
...SS2	Digital circuit ground (0V).
...TAL1	Input to the oscillator amplifier and to the internal clock generators if an external source is used.
...TAL2	Output of the oscillator amplifier.
...KOUT	Clock Output. The frequency of CLKOUT is programmable by a clock-divider-register. The Clock-Divider-Register (address FDH) defaults to divide by 2 after reset, but may be changed at run-time by the CPU to the required CLKOUT operating frequency. CLKOUT can sink/source four TTL/LS loads.

2.0 PIN DESCRIPTION (Continued)

Pin	Description
$\overline{CS}$	Chip select input pin. When this signal is low, the 82526 is selected by the CPU for transfer of command, status or data to or from the internal DPRAM, data to or from Port 0 and Port 1, or access to additional external devices within the address space 40H-EFH ($\overline{CS0}$–$\overline{CS2}$). $\overline{RD}$ or $\overline{WR}$ determine the direction of the data flow.
$\overline{INT}$	Interrupt output pin. An active low signal indicates to the CPU that the 82526 is requesting an interrupt. $\overline{INT}$ is an open drain output (external pull-up is required).
RX0, RX1	Serial data bus input pins to the on-chip differential input-comparator. RX0 is the "+" input and RX1 is the "−" input.
TX0	Serial data bus output pin from the on-chip programmable output-driver 0.
TX1	Serial data bus *output* pin from the on-chip programmable output-driver 1.
$\overline{RDY}$	Ready output pin used by the 82526 to slow the host that is accessing the DPRAM. RDY is an open drain output, pulled low when the 82526 is not ready.
ALE	Address latch enable input.
PORT 0	8-bit quasi-bidirectional I/O port (TTL/LS compatible).
PORT 1	8-bit quasi-bidirectional I/O port (TTL/LS compatible).
AD0–AD7	Multiplexed Address/Data Input/Output bus. AD0–AD7 are bidirectional three state lines connected to the system's Data/Address Bus for transfer of data, commands and status.
$\overline{CS0}$	Address decoding output. Chip select 0 pin is activated low if $\overline{CS}$ is set low and an address between 40H-BFH is output on the AD0–AD7 by the CPU.
$\overline{CS1}$	Address decoding output. Chip select 1 pin is activated low if $\overline{CS}$ is set low and an address between C0H-DFH is output on the AD0–AD7 by the CPU.
$\overline{CS2}$	Address decoding output. Chip select 2 pin is activated low if $\overline{CS}$ is set low and an address between E0H-EFH is output on the AD0–AD7 by the CPU.
$\overline{RST}$	An input for resetting the 82526. A low signal on this pin for 10 oscillator periods will cause the chip to terminate current operations and configure Port 0 and Port 1 as inputs. A Schmitt Trigger at this input pin is implemented for noise rejection. For normal operation, this pin should be high.

Pin(s)	Output Pin Status at Reset
Port 0	High with V-Pull-up
Port 1	High with V-Pull-up
Port 2	Float
CS0	High
CS1	High
CS2	High
READY	High (must remain high at Reset)
$\overline{INT}$	Float
XTAL2	Opposite of XTAL1
CLOCKOUT	Active
TX0	Float
TX1	Float

NOTE:
$\overline{CS0}$, $\overline{CS1}$ and $\overline{CS2}$ are push-pull outputs, and may be used to select additional system peripherals.

6

3.0 FUNCTIONAL DESCRIPTION

3.1 DPRAM Address Map

The buffer memory area used for communication is configured by the user during an initialization download after power-up. It consists of dedicated control registers, communication objects, and an end mark signifying the end of the DPRAM area used for communication. Each individual communication object is made of the identifier, control bits, and data. The CPU operates only on this communication buffer to perform message transfers; the "Bus Interface Logic" of the 82526 manages the bus traffic. The buffer memory layout is shown in Figure 3.

Addr.	Function	Comments	Default Value at Reset
00H	Control-Register	Global Status and Control Registers	Random Value
01H	Status-Register		
02H	Interrupt-Pointer		
03H	Bus Timing-Register 0		
04H	Bus Timing-Register 1		
05H	Output Control-Register		
06H 07H 08H 09H	Descriptor 1 Descriptor 2 Descriptor 3 Data-Segment 1/Byte 1 Data-Segment 1/Byte N	Communication Object 1	Random Value
	Descriptor 1 Descriptor 2 Descriptor 3 Data-Segment 2/Byte 1 Data-Segment 2/Byte M	Communication Object 2	Random Value
Max Address 3EH	End Mark FF (Hex)	End of Message Objects	Random Value
3FH	User Segment (After End Mark)	Possible RAM Extension For User Programs	Random Value
40H BFH	Decoded by $\overline{CS0}$	Chip Select Output 0	No Reg Value
C0H DFH	Decoded by $\overline{CS1}$	Chip Select Output 1	No Reg Value
E0H EFH	Decoded by $\overline{CS2}$	Chip Select Output 2	No Reg Value
F0H FCH	Not Used		No Reg Value
FDH	Clock Divider Register	CLKOUT Pin	00H*
FEH	Port 0 Data Register	Port 0 Input/Output	FFH**
FFH	Port 1 Data Register	Port 1 Input/Output	FFH**

*Write Only Register reads back FFH only.
**Drawn high by weak pull-up if no external device is connected.

Figure 3. DPRAM Address Map

3.2 Control Register (Address 00H)

Reset Request

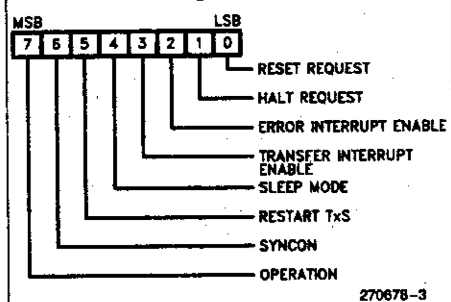
RESET REQUEST is set by the CPU and read by the IMP. When set to "high" and detected by the IMP, it causes an SIU (Serial Interface Unit) internal hardware reset. The execution of the reset request is acknowledged to the CPU via the reset status bit (see Status Register), set by the IMP indicating to the host that the 82526 is ready for initialization. Thereafter, the IMP remains in a "loop mode" waiting for the reset request to be set "low" by the host after completion of the DPRAM configuration. The 82526 returns to the operating mode immediately after the reset status bit is set "low" by the IMP. During the reset sequence, the PIU, clock generator, and the buffer memory remain active.

NOTE:
For proper operation, the reset request bit must be set to "high" for any start-up or new DPRAM configuration routine and set to "low" after the configuration.

Reset will most often be used the first time after power-up or after the 82526 has removed itself from the CAN Bus. (busoff state; see Section 7.7 for details.)

Reset request set "high" does not break a transmission or reception of a message in process, but will stop the 82526 from transmitting or receiving the next message.

Control Register: Address 0H



Once reset is requested by the CPU, the reset status should be confirmed by the CPU before any changes are made to the DRAM.

Halt Request

HALT REQUEST is set by the CPU and read by the IMP. When set to "high" the 82526 continues processing the current transmission or reception and then stops any further activity until the halt request bit is set to "low" which restarts normal operation. Halt request forces the IMP to stay in an idle-loop while the PIU, SIU and clock generator remain active. Halt Request is used to reconfigure communication objects while maintaining error history. Reset Request can also be used to reconfigure communication objects but it clears all error history (often used at initialization).

Error Interrupt Enable

ERROR INTERRUPT ENABLE is set by the CPU and read by the IMP. If set to "high", the interrupt output signal of the 82526 to the CPU is enabled, and disabled if set to "low". The 82526 may generate an error interrupt for the following reasons:

— Error status bit set (Status Register)
— Bus status set to "off bus" (Status Register)
— DPRAM status set to "error" (Status Register)

Also see Error Confinement in this Overview.

Transfer Interrupt Enable

TRANSFER INTERRUPT ENABLE is set by the CPU and read by the IMP. If set to "high", the 82526 will generate an interrupt to the CPU after a communication object is successfully received or transmitted with the transfer interrupt enable bit within the Descriptor Byte 3 of the corresponding communication object set. If set to "low", any transfer interrupt to the CPU is disabled independent of the transfer interrupt enable bit within the descriptor of the communication object.

Sleep Mode

SLEEP MODE is set by the CPU and read by the IMP. If the bit is set to "low" (normal operation) the 82526 will not enter the sleep mode. If set to "high", the device will enter sleep mode when serial bus is idle for at least 256 BIT-TIMES (defined in Section 3.5.1) and the CPU has not accessed the 82526 during the same length of time. This allows sufficient time for the IMP to terminate current processes. A return from sleep mode to the normal operation will be activated by the following events:

— Start bit detection on the serial bus
— Any CPU access to the DPRAM, Port 0, Port 1, or the clock divider register.

Sleep mode is transparent to the user, peripherals and other nodes. An attached host CPU does not see that the 82526 is in the sleep mode. Hence, all timings for DPRAM and I/O access remain valid for the sleep mode. The crystal remains active during Sleep Mode in order to implement an instantaneous "wake-up".

6

rt TxS

ART TRANSMIT SEARCH). Any write access CPU to set this bit "high" forces the IMP to new transmit search loop at the first Communi- Object. *Placing the highest priority message at the the DPRAM and using the Restart TxS bit gives a m latency time for the object being placed on the* is not necessary to reset this bit between subse- "restart transmit search" requests.

on

ON is set by the CPU and read by the IMP. This trols the type of resynchronization of the Bit Logic during the reception of a frame. If set to ", bus line transitions from "recessive" to "domi- as well as from "dominant" to "recessive" are or resynchronization.

to "low", only transitions from "recessive" to nant" are used for resynchronization.

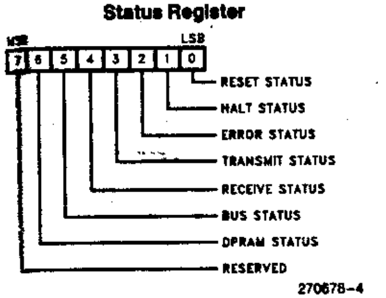
NOTE:
ot recommended to resynchronize on both tran- at bit rates greater than 100K bits per second. see Section 3.5.7 on synchronization.)

ation

ATION bit must be set to "low" by the CPU at ation. This will force the 82526 to start opera- s soon as the reset status bit is set low by the IMP. peration bit must never be set to "high" by the because it will result in an unpredictable mode.

Status Register (Address 01H)

Status Register



270678-4

The Status Register is modified by the IMP only. After power up reset and/or reset request from the CPU, the IMP will set the reset status bit "high". It is set "low" after the reset request is released by the CPU. Even though the CPU can write this register at any time, CPU should only clear this register during initialization. After start up, a read access to this register will report the status of the 82526.

Reset Status

RESET STATUS bit set "high" by the IMP acknowledges to the CPU that the reset request from the CPU was detected and an internal hardware reset has been performed by the 82526. The bit is set back to "low" by the IMP after the CPU has set the Reset Request Bit to "low"; this forces the 82526 to start or restart normal operation.

Halt Status

HALT STATUS bit is altered corresponding to the Halt Request from the CPU. It is set "high" by the IMP to indicate that the Halt Status (Idle Loop) was entered, and set "low" when the Halt Request is released by the CPU. If Halt Request is released when the Serial Interface Unit (SIU) is in the reception mode, the IMP remains in a wait loop until the next interframe space time. This bit must be monitored by the CPU to confirm the Halt Status prior to changing the DPRAM configuration.

Error Status

ERROR STATUS bit is set "high" by the IMP to indicate that the Error Management Logic (EML) has detected a problem either with transmission or reception of messages. It also indicates that the Transmit Error Counter (TEC) or Receive Error Counter (REC) has reached a value of 96. The EML monitors receive and transmit errors and takes appropriate actions (see error confinement for more details). The Error Status bit is set "low" during normal operation by the IMP.

Transmit Status

TRANSMIT STATUS bit set "high" by the IMP indicates to the CPU that the 82526 currently is in the transmit mode. A "low" is set by the IMP when either 1) the 82526 enters the idle mode after a successful transmission, or 2) there was a loss of the arbitration during transmission of the message identifier forcing the 82526 to immediately enter the receive mode, or 3) because an error was detected during transmission.

Receive Status

RECEIVE STATUS bit set "high" by the IMP to indicate to the CPU that the 82526 is in the receive mode. It is set to "low" after reception is complete.

NOTE:
The 82526 is in the idle mode if both, the Transmit and the Receive Status bits, are set "low". In this case, the bus is idle and the IMP has not found a Communication Object to transmit.

Bus Status

BUS STATUS is modified by the IMP and read by the CPU. If set to "low", the 82526 is bus active (on - bus). If set to "high" (off - bus), the 82526 remains in the bus-off mode until a Reset Request is set by the CPU and executed by the IMP and Reset Request is released by the CPU. It also indicates that the Transmit Error Counter (TEC) has reached a value of 255. Also see Reset Request Bit and Error Confinement for additional details.

DPRAM Status

DPRAM STATUS bit is normally set to "low" by the IMP after a complete DPRAM configuration is set up by the CPU and no buffer memory configuration inconsistency is detected by the IMP. The bit will be set to "high" if a buffer-memory inconsistency (faulty communication object) is detected after the 82526 has started operation. The inconsistency might be: (a) if the end mark is not before 3FH, or (b) the data length code and the number of data bytes in a communication object do not correlate.

3.4 Interrupt Pointer Register (Address 02H)

270678-5

The Interrupt Pointer is an 8-bit register that contains:

(a) address of the status register (01H) if an interrupt has occurred because of an error. This enables the user to determine the source of the error,

(b) address of a communication object for which transfer status is complete ("high"),

(c) an empty pointer (FFH), otherwise.

The CPU acknowledges an interrupt from the 82526 by writing empty pointer to the interrupt register. This resets the $\overline{INT}$ pin. A read of the interrupt pointer register by the CPU resets the $\overline{INT}$ pin but is not an acknowledge.

3.5 Bus Timing Logic (BTL)

The BTL monitors the serial bus line via its input comparator and performs all the bus line related bit timing. The baud rate prescaler also is a part of the BTL.

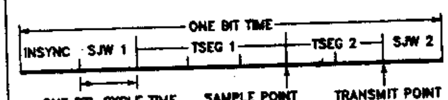
The BTL performs the following tasks:

- monitoring of the serial bus
- adjustment of the sample point within the bit time (programmable)
- synchronization to the bit stream
  - hard synchronization at start of frame
  - resynchronization during a frame transfer in order to compensate differences of transmitter/receiver clock frequencies of individual 82526s
- majority decisions on the bit level (programmable)
- prevention of nonsynchronization (spikes)

The configuration of the BTL is done during the initialization of the 82526. There are three registers in the 82526 communication buffer DPRAM which relate to the BTL: CONTROL-REGISTER, BUS-TIMING-REGISTER 0 and BUS-TIMING-REGISTER 1.

3.5.1 BIT TIMING

A bit time is subdivided into a number of BTL cycles. This number results from the addition of the segments SJW 1, INSYNC, SJW 2, TSEG 1 and TSEG 2.



270678-6

MEANING OF THE SEGMENTS

INSYNC

The incoming edge of a bit is expected during this state; this segment corresponds to one BTL cycle.

SJW 1 and SJW 2 Sychronization

Synchronization Jump Widths are used to compensate for phase shifts between clock oscillators of different bus nodes.

6

segments (SJW 1 and SJW 2) determine the maxi- jump width for resynchronization and are pro- nable from 1 to 4 BTL cycles. The width of SJW icreased to a maximum of twice the programmed during resynchronization. The width of SJW 2 is ed or canceled to shorten the bit time during re- ronization.

ceive mode. The transmit point is determined internally in such a way that, with zero delay, the generated transmit signal will appear within the INSYNC state.

Additional programmable segments are SAM, SYN-CON and the baud rate. See control registers for details.

ɜ 1

sampling point is based on the number of BTL programmed by TSEG 1 (4 bits). The sampling is located at the end of TSEG 1 (SAM = 0). 1 is used to compensate delay times on the bus reserve time to tolerate one or more mis-syn- ization pulses caused by spikes on the bus line. 1 is programmable from 1 to 16 BTL cycles.

ɜ 2

defines the time between the sampling point and ransmit point, programmable from 1 to 8 BTL cy-

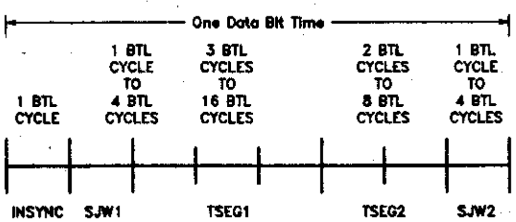
segment is necessary to tolerate one or more mis- ronization spikes on the bus line. It is also neces- to guarantee sufficient time for BSP to analyze the le taken from the bus and decide if it has lost ration. When this happens, the transmit logic im- ately stops the transfer and the SIU enters the re-

3.5.2 BIT TIME CALCULATION

Baud Rate Prescaler(1)	$= 2^4 \cdot BRP4 + 2^3 \cdot BRP3 + 2^2 \cdot BRP2 + 2^1 \cdot BRP1 + BRP0$
1 BTL Cycle	= 2 * (Baud Rate Prescaler + 1) * XTAL Cycle
1 System cycle	= 2 * XTAL Cycle
1 Bit Cycle	= (INSYNC + SJW1 + TSEG1 + TSEG2 + SJW2) BTL Cycles
	= N * XTAL Cycles
Baud Rate	= FREQ_XTAL/N

NOTE:
1. Baud Rate Prescaler is defined by Bus Timing Register (03H).

This Figure shows timing for one Data Bit time on the 2-Wire Bus:



270678-7

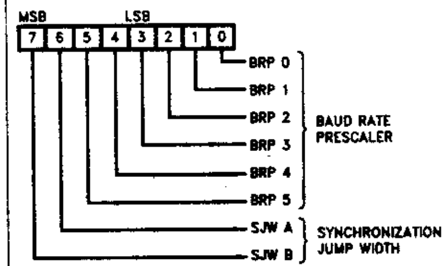
3.5.3 BUS TIMING REGISTERS

Bus Timing Register 0 and Register 1 are located at 03H and 04H respectively in the DPRAM.

3.5.3.1 Bus Timing Register 0

Bus Timing Register 0 is used for programming the baud rate prescaler and synchronization jump width.

Bus Timing Register 0 (Add. 03H)



270678-8

Baud Rate Prescaler (BRP)

By programming the six bits of the baud rate prescaler, the BTL cycle time is determined. The BTL cycle time is derived from the system cycle time (the system cycle time is twice the crystal cycle time). The desired baud rate is determined by the BTL cycle time and the programmable bit timing segments.

Baud Rate Prescaler Table

Baud Rate Prescaler BRP 0 - BRP 5	BTL Cycle Time
000000	1 × System Cycle Time
000001	2 × System Cycle Time
000010	3 × System Cycle Time
:	:
:	:
111111	64 × System Cycle Time

Synchronization Jump Width (SJW)

The Synchronization Jump Width defines the maximum number of BTL cycles that a bit may be shortened or lengthened by one resynchronization during transmission of a Data Frame or Remote Frame. Synchronization Jump Width is programmable by bits SJWB and SJWA as shown in the table.

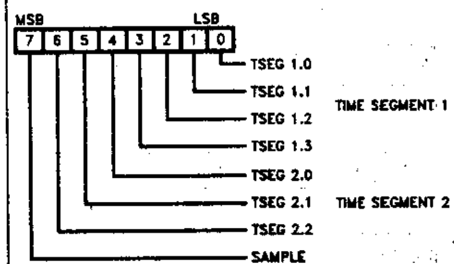
Synchronization Jump Width Table

SJWB	SJWA	SJW1 or SJW2
0	0	1 BTL Cycle
0	1	2 BTL Cycles
1	0	3 BTL Cycles
1	1	4 BTL Cycles

3.5.3.2 Bus Timing Register 1

Bus Timing Register 1 is used for programming the number of samples per bit, Delay Time and the Phase Shift Buffer.

Bus Timing Register 1 (Add. 04H)



270678-9

SAMPLE: This determines the number of samples of the serial bus which are taken by the BTL. If SAMPLE is set to "low", a bit is sampled only once, if set to "high", three samples per bit are taken. Sample = 0 allows higher bit rate while Sample = 1 gives better rejection to noise on the bus. In addition, setting Sample = 1 will increase the bit time by 1 BTL cycle.

6

Time Segment 1 and Time Segment 2 (TSEG1, TSEG2)

TSEG1 and TSEG2 are programmable as shown in the tables below:

TSEG				TSEG1
1.3	1.2	1.1	1.0	
0	0	0	0	1 BTL Cycle
0	0	0	1	2 BTL Cycles
0	0	1	0	3 BTL Cycles
0	0	1	1	4 BTL Cycles
:	:	:	:	: : :
1	1	1	1	16 BTL Cycles

TSEG			TSEG2
2.2	2.1	2.0	
0	0	0	1 BTL Cycle
0	0	1	2 BTL Cycles
:	:	:	
1	1	1	8 BTL Cycles

Example 1: Calculation of bit rate

Given: Reg 03H = 0H, Reg 04H = 12H, Crystal = 16 MHz

Reg 03H

SJW		BRP					
B	A	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Reg 04H

Sample	TSEG			TSEG			
	2.2	2.1	2.0	1.3	1.2	1.1	1.0
0	0	0	1	0	0	1	0

Baud Rate Prescaler = 0 (i.e., BTL Cycle = System Cycle)

In One Bit Time one sample is taken from the serial bus.

INSYNC	1 BTL cycle
SJW1	1 BTL cycle
TSEG1	3 BTL cycles
TSEG2	2 BTL cycles
SJW2	1 BTL cycle
Total	8 BTL cycles

System Clock Freq = Crystal Frequency / 2

8 BTL cycles * 2 XTAL cycles = 16 XTAL cycles per data bit

16 MHz crystal / 16 XTAL cycles per data bit = 1 Mbit/sec

Example 2: Calculation of bit rate

Given: Reg 03H = 40H, Reg 04H = 33H, Crystal = 16 MHz

Reg 03H

SJW		BRP					
B	A	5	4	3	2	1	0
0	1	0	0	0	0	0	0

Reg 04H

Sample	TSEG			TSEG			
	2.2	2.1	2.0	1.3	1.2	1.1	1.0
0	0	1	1	0	0	1	1

Baud Rate Prescaler = 0 (i.e., BTL Cycle = System Cycle)

INSYNC	1 BTL cycle
SJW1	2 BTL cycles
TSEG1	4 BTL cycles
TSEG2	4 BTL cycles
SJW2	2 BTL cycles
Total	13 BTL cycles

System Clock Frequency = Crystal Frequency / 2

13 BTL cycles * 2 XTAL cycles = 26 XTAL cycles per data bit

16 MHz crystal / 26 XTAL cycles per data bit = 615 Kbit/sec

3.5.4 BTL CONFIGURATION REQUIREMENTS

Special requirements for the configuration of the BTL relate to the location of the sample point.

The correct location of the sample point is very important for proper function of a transmission, especially at high speed and maximum cable length. For this reason, the following items need to be considered:

— At start of frame, all 82526s in the system synchronize "hard" on the first recessive to dominant edge start bit. During arbitration, however, more than one node may transmit in coincidence. As a result, it may take two times the bus delay plus the time of the output driver and the input comparator until the bus line is stable. Correspondingly, the duration of TSEG 1 should reflect at least the total delay time.

— To improve the behavior with respect to spikes on the bus line, it is recommended to have an additional synchronization buffer on the left and right side of the sample point to allow one or more nonsynchronizations without sampling the wrong position within a bit time. At a minimum, this buffer should correspond to the time of the SJW segments.

3.5.5 CONFIGURATION RESTRICTIONS

A. Restriction related to the location of the transmit point.

Restrictions on the configuration are based on internal processing timing and relate to the location of the transmit point.

The transmit point is generated automatically, but the following conditions must be noted:

— After the sample point, the actual bus level is known. Dependent on the bus value, any node decides either to start, stop, or continue a transmission. The decision requires a minimum of two BTL cycles for internal processing time. After this time period, the 82526 is ready to transmit.

— The start point to transmit a new bit occurs earlier than the bit actually monitored on the bus. This is to compensate the internal delay times such as synchronizing to the internal clock or to perform the majority process (SAM = 1).

Corresponding to the issues, the following restrictions regarding programming TSEG 2 and SJW1/SJW2 apply:

for SAM = 0: TSEG 2 + SJW 2 ≥ 3 clock cycles minimum

for SAM = 1: TSEG 2 + SJW 2 ≥ 4 clock cycles minimum

When Baud Rate Prescaler is programmed to zero and:

(a) sample = 0, TSEG1 must be a minimum of 3 BTL cycles

(b) sample = 1, TSEG1 must be a minimum of 4 BTL cycles.

NOTE:
This restriction is valid only if the baud rate-prescaler is programmed to zero.

B. Restriction related to the Bit Stream Processor (BSP)

Because of internal processing time, it must be ensured that two successive sample points never get closer than a minimum of five BTL cycles even when shortened by hard synchronization or resynchronization. In the most critical case, the segments TSEG 2 and SJW 2 are shortened by a hard synchronization.

SJW 1 + TSEG 1 ≥ 4 BTL cycles min

NOTE:
This restriction is valid only if the baud rate-prescaler is programmed to zero.

C. Restriction related to the Bit Timing Logic (BTL)

Because of BTL internal requirements, it is not acceptable to program the following parameters at the same time:

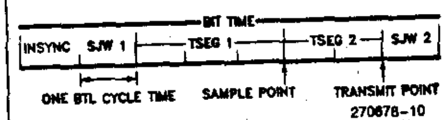
TSEG 1 = 0 and SJW 1 = 0 or

TSEG 1 = 0 and SJW 1 = 1

NOTE:
This restriction is independent of the value of the baud rate-prescaler.

3.5.6 MINIMUM CONFIGURATION OF BTL BAUD RATE-PRESCALER = 0

Considering all restrictions, a bit time can be programmed to a minimum of eight BTL cycles for baud rate-prescaler = 0 and SAM = 0.



270678–10

NOTE:
One BTL Cycle = One System Cycle, for BRP = 0H.

Considering all restrictions, a bit time can be programmed to a minimum of six BTL cycles for baud-rate-prescaler ≥ 1 and SAM = 0.

3.5.7 SYNCHRONIZATION

Synchronization is performed by a state machine which compares the incoming edges with its actual bit timing and adapts the bit timing by hard synchronization or resychronization.

6

RD SYNCHRONIZATION occurs only at the be-ng of a frame. The 82526 synchronizes to the first ...ning recessive to dominant edge start bit of a ...me

SYNCHRONIZATION occurs during the message stream to compensate for small differences in the ...llator frequencies of individual 82526s as well as ...nges introduced by switching from one to another ...smitter; e.g., after arbitration of two or more nodes.

...W 1 and SJW 2 define the maximum number of clock ...les a bit time may be shortened or lengthened by ...synchronization.

...synchronization can be performed on both edges—...essive to dominant and dominant to recessive or on ...recessive to dominant edge only, dependent on the ...NCON bit in the Control Reg (Add. 0H).

...NCON = 1: Synchronization on both edges. Syn-...nization is always done on the edge of a bus level ...ch is different from the one read at the last sample ...int.

...NCON = 0: Synchronization on the edge of a domi-...nt level only if the bus level monitored at the last ...mple point was a recessive level.

...is restriction is necessary because of physical bus ...aracteristics at the maximum cable length and high ...ud rates.

...nchronization is done once during a bit time and is ...eased after the sample point dependent on the level ...the actual bit. Thereafter, the BTL state machine ...nchronizes to the next relevant edge and then, syn-...ronization is being suppressed until the following ...mple point.

...resynchronization, the bit time can be lengthened or ...ortened based on the following conditions:

- ...an edge appears in the segment SJW 1 or TSEG 1 → lengthen
- ...an edge appears in the segment SJW 2 or TSEG 2 → shorten

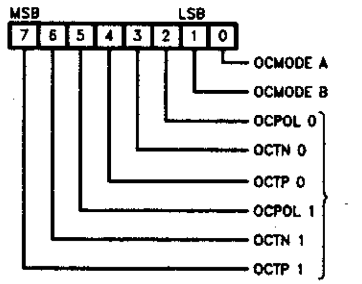
NOTE:
...82526 transmitting a dominant bit does not ...ngthen (synchronize) the bit time if SYNCON is set ...zero. This is important for a high speed bus in or-...r to ensure the delay time of the output driver and ...put comparator does not cause a permanent length-...ing of the bit time of the transmitter and, conse-...ently, of all receivers.

3.6 Output Control Circuitry and Input Comparator

3.6.1 OUTPUT CONTROL CIRCUITRY

The Output Control Register allows set up of different output driver configurations under software control. The user may select active pull-up, pull-down, float, or push-pull output.

Output Control Register (Add. 05H)



270678-11

PROGRAMMABLE FEATURES

Table 1. Programmable Output Functions

OCMODE B	OCMODE A	Function
1	0	Normal Mode Bit stream transmitted on both, Tx0 and Tx1
1	1	Normal Mode Tx0: Bit Stream Tx1: Bus Clock (txclk)
0	0	Bipolar Mode
0	1	Not Used

OUTPUT DRIVER CONFIGURATION

OCPOL i refers to the polarity of Transmit Pins TX i (i = 0, 1)

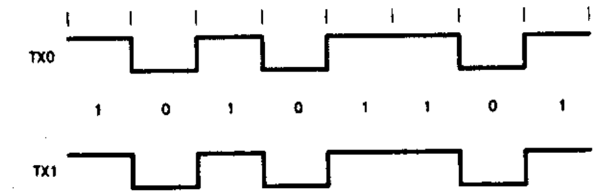
OCTN i refers to the pull down mode of the TX i as active

OCTP i refers to the pullup mode of the TX i as active

When both OCTN i and OCTP i of TX i (i = 0, 1) are active, it refers to the push pull mode.

When only OCTN i is active, it refers to pull down

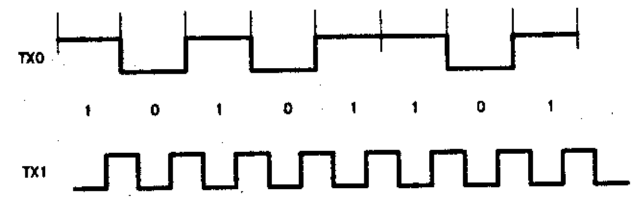
Normal Mode: OCMODE A = 0, OCMODE B = 1, Data = 10101101



270678-12

NOTE:
TX0 and TX1 output are the same.

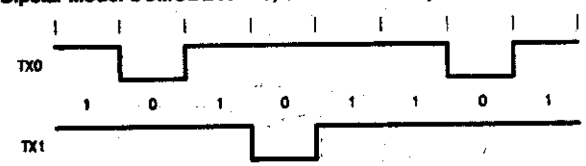
Normal Mode: OCMODE A = 1, OCMODE B = 1, Data = 10101101



270678-13

NOTE:
TX0 = Data
TX1 = Clock based on BTL parameters.

Bipolar Mode: OCMODE A = 0, OCMODE B = 0, Data = 10101101



270678-14

NOTE:
Dominate bit alternates between TX0 and TX1 for zero D.C. offset.

6

When only OCTP i is active, it refers to pull up.

Table 2. Tx0, Tx1 Output Driver Configurations

Mode	OCTPi	OCTNi	OCPOLi	TD	TPi	TNi	TXi Output Level
FLOAT	0	0	0	0	off	off	float
	0	0	0	1	off	off	float
	0	0	1	0	off	off	float
	0	0	1	1	off	off	float
PULL DOWN	0	1	0	0	off	on	low
	0	1	0	1	off	off	float
	0	1	1	0	off	off	float
	0	1	1	1	off	on	low
PULL UP	1	0	0	0	off	off	float
	1	0	0	1	on	off	high
	1	0	1	0	on	off	high
	1	0	1	1	off	off	float
PUSH PULL	1	1	0	0	off	on	low
	1	1	0	1	on	off	high
	1	1	1	0	on	off	high
	1	1	1	1	off	on	low

NOTE:
TD = data bit to be transmitted; 0 = dominant
1 = recessive
TPi/TNi = on-chip transistors
TXi = Tx0, Tx1 Serial Output Pins

The output control circuitry can also be described logically, as in Figure 4.

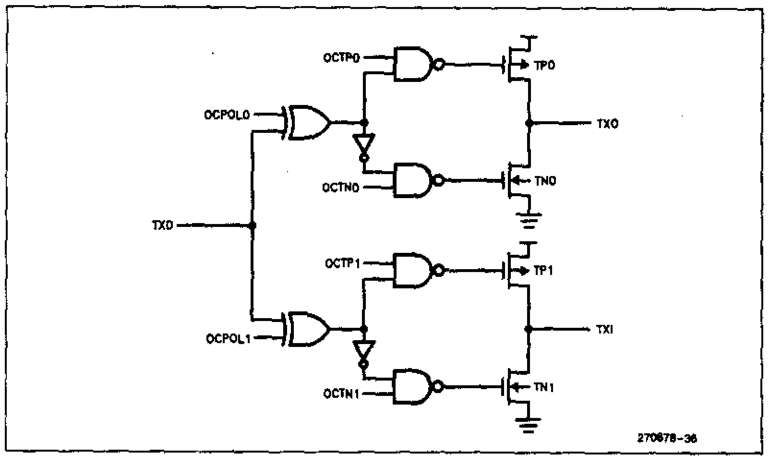


Figure 4. Logical Description of the Output Control Circuitry

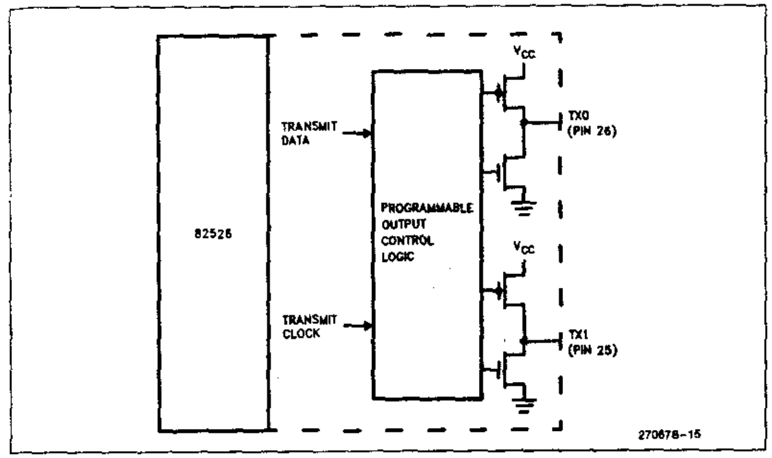


Figure 5. Output Control Logic of 82526

Normal Mode

Output signal levels are assigned to the logical bit levels "dominant" and "recessive" depending on the output characteristics (pull-up, pull-down, push-pull).

Bipolar Mode

If galvanically decoupled from the busline by a transformer, the bit stream has to be coded in such a way that there is no resulting dc-current transferred from the transformer.

With "recessive" bits, all outputs are deactivated. "Dominant" bits are sent alternately on Tx0 and Tx1; e.g., the first "dominant" bit is sent on Tx0, the second on Tx1, the third on Tx0 and so on.

3.6.2 INPUT COMPARATOR

The 82526 facilitates differential data transmission through the Input Comparator. The input comparator is shown in Figure 6.

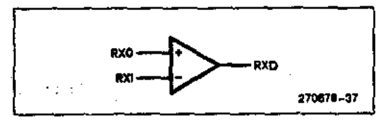


Figure 6. Input Comparator Showing Polarities

3.7 Communication Object(s)

Communication Objects are created for every message by the user to perform a data tranfer. This allows message transfer and an integrity check to be handled without user interference. According to Figure 3 (Buffer Memory Layout), each Communication Object consists of three Descriptor Bytes and a data segment.

The number of Communication Objects which may be configured within the buffer memory is dependent on the number of bytes in the data field (Data Segment) since Communication Objects are placed contiguously.

DPRAM size is 64 registers. Registers 0 to 5 are used for general setup and control. Each Communication Object requires three Descriptor Bytes Registers and may use zero to eight Data Bytes. The End Mark requires one register. The maximum address for the end mark is 3EH (due to IMP operating constraints). Any DPRAM bytes from the End Mark to address 3FH may be used as general RAM by the controller.

If the user was transmitting or receiving only one byte of data, eleven registers would be required. Six Control Registers, three Descriptor Bytes, one Data Byte and an End Mark. The additional registers could be used as RAM memory by the controller.

Control Registers	(6)
Descriptor Registers	(3)
Data Byte	(1)
End Mark	(1)
Total Registers	11

Address	Function	Comment
00H	Control Register	Six Control Registers
01H	Status Register	
02H	Interrupt Pointer	
03H	Bus Timing Register 0	
04H	Bus Timing Register 1	
05H	Output Control Register	
06H	Descriptor Byte 1 Object 1	Three Descriptor Bytes
07H	Descriptor Byte 2 Object 1	
08H	Descriptor Byte 3 Object 1	
09H	Data Byte 1	Data Byte
0AH	End Mark (FFH)	End Mark
0BH	Free User RAM after End Mark	User RAM
3FH	Last User RAM before $\overline{CS0}$	User RAM

If eight bytes of data were to be transmitted or received, eighteen registers would be required. Six Control Registers, three Description Bytes, eight Data Bytes and an End Mark are shown below:

Control Registers	(6)
Descriptor Registers	(3)
Eight Data Bytes	(8)
End Mark	(1)
Total Registers	18

Address	Function	Comment
00H	Control Register	Six Control Registers
01H	Status Register	
02H	Interrupt Pointer	
03H	Bus Timing Register 0	
04H	Bus Timing Register 1	
05H	Output Control Register	
06H	Descriptor Byte 1 Object 1	Three Descriptor Bytes
07H	Descriptor Byte 2 Object 1	
08H	Descriptor Byte 3 Object 1	
09H	Data Byte 1	Eight Data Bytes
0AH	Data Byte 2	
0BH	Data Byte 3	
0CH	Data Byte 4	
0DH	Data Byte 5	
0EH	Data Byte 6	
0FH	Data Byte 7	
10H	Data Byte 8	
11H	End Mark (FFH)	End Mark
12H	Free User RAM after End Mark	User RAM
3FH	Last User RAM before $\overline{CS0}$	User RAM

If more than one message object is placed in the DPRAM the six control bytes are not duplicated.

The following example shows six message objects with the Control Registers starting at 00H and the End Mark at location 3EH.

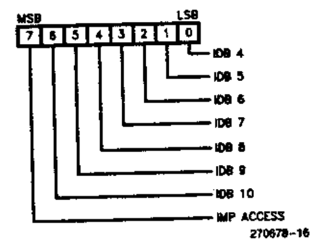
Control Registers	(6)
Descriptor Registers	(3) Object 1
Eight Data Bytes	(8)
Descriptor Registers	(3) Object 2
Eight Data Bytes	(8)
Descriptor Registers	(3) Object 3
Eight Data Bytes	(8)
Descriptor Registers	(3) Object 4
Eight Data Bytes	(8)
Descriptor Registers	(3) Object 5
Eight Data Bytes	(6)
Descriptor Registers	(3) Object 6
End Mark	(1) End Mark
Total Registers	(63) (00H - 3EH)

NOTE:
Object 6 has 0 Data Bytes. This is useful for Timing Sync with external real world signals.

Descriptor Bytes 1, 2 and 3

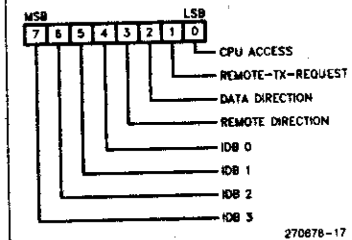
The Descriptor consists of three bytes, assigned to each Communication Object by the user. The three bytes include the message identifier and a control segment which contains semaphore bits to guarantee mutually exclusive access by the CPU and the IMP to the Communication Object, if required.

Descriptor Byte 1



270678-16

Descriptor Byte 2

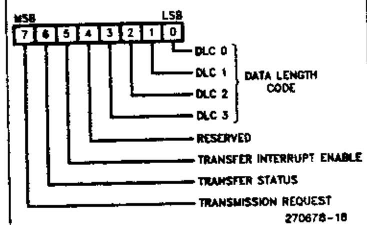


270678-17

Identifier

An Identifier is a unique name for each Communication Object and defines the corresponding, specific message. It also determines its priority for bus access (see arbitration).

Descriptor Byte 3



270678-18

The Identifier consists of eleven bits (IDB 0 to IDB 10). IDB 10, the most significant bit is transmitted first following the start bit.

NOTE:
IDB 10 to IDB 4 set to "high" cannot be used in an Identifier because 0FFH also serves as End Mark. This results in a total number of 2032 different Identifiers (messages), starting from 00000H (highest priority) to 07EFH (lowest priority).

Control Bits

IMP ACCESS is set by the CPU and read by the IMP. If set to "low", IMP access to the data of this Communication Object is "locked". If reset to "high", IMP access is released.

6

DATA SEGMENT holds the data of the corresponding Communication Object. The number of data bytes per segment is defined by the data length code in each descriptor.

ENDMARK set to FFH by the CPU indicates to the IMP that there is no Communication Object stored beyond the Endmark (max. address 3EH). The Endmark indicates to the IMP the end of the linked list of Communication Objects. A Communication Object Identifier cannot have FFH for the most significant bits as it would conflict with the Endmark.

USER SEGMENT may follow the Endmark since it is not used by the 82526 controller. It is available to the user as general data memory and does not affect the function of the 82526.

4.0 82526 FRAME TYPES

The 82526 communication controller supports four different frame types:

— data frame
— remote frame
— error frame
— overload frame

4.1 Data Frame

A Data Frame is composed of seven different fields:

— start bit
— arbitration field
— control (identifier) field
— data field (data segment)
— CRC field
— acknowledge field
— end of frame

START OF FRAME signals the start of a data- or remote frame. It consists of a single dominant bit used for synchronization of receiving nodes.

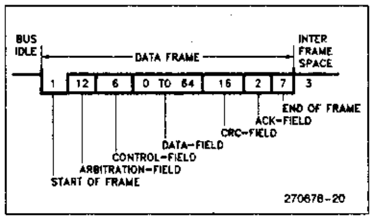


Figure 7. Data Frame Format

ARBITRATION FIELD consists of the message Identifier and one additional control bit (Remote Transmission Request).

The simultaneous message transmission start of two or more nodes is solved by bitwise arbitration during the transmission of the Arbitration Field (for more details see Arbitration).

After the arbitration field there is only one transmitter on the serial bus. Since the arbitration is performed only in the arbitration field, two different nodes must not use the same ID to transmit a Data Message.

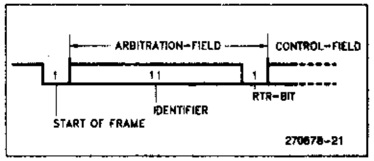


Figure 8. Arbitration Field Format

IDENTIFIER: This 11-bit field provides information about each individual message and in priority. The Identifier defines the corresponding specific message (e.g., engine speed, temperature, pressure, etc.) of a Communication Object. Its digital value represents the message priority for bus access. The lower the ID the higher the priority.

NOTE:
An Identifier does not define a physical station address to determine a Receiver of a frame on a multidrop line. The 82526 controller decides, based on the acceptance filter process of a received Identifier, whether data being received within a correct frame are to be accepted or not.

Within an 82526 based communication network there is no discrimination between a point-to-point, multicast or broadcast communication.

RTR BIT: A station, active as a receiver may initiate the transmission of data from another node, by transmission of a Remote Frame to the network, addressing the data source via the Identifier. Within a data frame, the RTR-bit is transmitted as a dominant bit level (for more information see Remote Frame).

82526 ARCHITECTURAL OVERVIEW

CONTROL FIELD: This field consists of six bits. It includes the data byte count and two reserved bits (following the RTR-bit) which are automatically transmitted with a dominant bit level. The length of the data field (data segment) is coded in bytes. A "0-bit" in the data length code (descriptor byte 3, DCL0–DCL3) is transmitted as a dominant level and a "one-bit" as a recessive level.

DATA FIELD: Data stored within the corresponding Data Segment in the Communication Buffer are transmitted within the data field. The length of the data field varies from 0 to 8 bytes based on the value of the data byte count (data length code). The byte at the lowest address of the data segment will be transmitted first with the MSB sent first.

CRC FIELD contains the CRC check sum (15 bits) followed by the CRC delimiter (1 bit). The cyclic redundancy code includes the start bit (start of frame), arbitration field, control field, data field and CRC field. The most significant bit (MSB) is transmitted first. The frame check sequence implemented in the 82526 is derived from a cyclic redundancy code best suited for frames with a total bit count of less than 127 bits (BCH-Code).

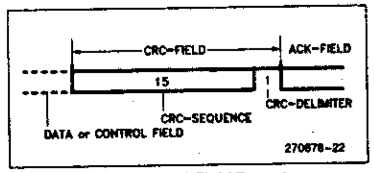


Figure 9. CRC Field Format

ACKNOWLEDGE FIELD: The ACK-FIELD consists of two bits, the ACK-SLOT and ACK-DELIMITER; the transmitter sends a "recessive" level for these bits. Any receiving 82526 acknowledges to the transmitting 82526 a match of the complete/correct CRC check sum within the ACK-SLOT by superscribing the recessive bit with a dominant bit. A transmitter monitoring the bus level recognizes that at least one receiver within the system has received a complete and correct message (no error was found).

The ACK-DELIMITER (recessive bit) is the second bit of the ACK-FIELD. As a result, the ACK-SLOT is surrounded by two recessive bits—the CRC delimiter and the ACK delimiter.

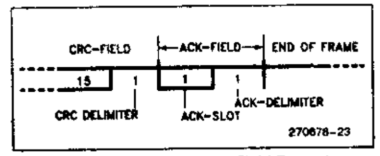


Figure 10. Acknowledge Field Format

END OF FRAME: Each Data Frame or Remote Frame is delimited by the end of frame bit sequence which consists of seven "recessive" bits (exceeding the bit stuff width by two bits). A receiver detects the end of a data frame independent of a previous transmission error because the receiver expects all bits up to the end of CRC-FIELD to be coded by the bit stuffing method. Bit stuffing is not used in end of frame.

4.2 Remote Frame

A Remote Frame is composed of six different fields:

— start of frame
— arbitration field
— control field
— CRC field
— ACK field
— end of frame

Contrary to the Data Frame, the RTR-bit of the Remote Frame is "recessive" and no data segment is transmitted independent of the Data Length Code set by the Descriptor of the corresponding Communication Object.

The RTR-bit allows Remote Transmission Requests from any node to the system. This provides the capability to request information in addition to the standard broadcast characteristics. It also supports powerful diagnostic capability by being able to determine if the primary transmitter (data source) of a specific parameter(s) is on the bus and functional.

4.3 Error Frame

The Error Frame contains a sequence of variable length dominant bits as a result of error flags being transmitted by different system-nodes. This is an important aspect of the 82526 communication protocol with regards to data consistency within a communication network. The error frame is followed by an error delimiter.

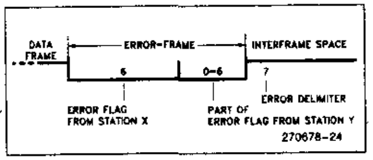


Figure 11. Error Frame Format

ERROR FLAG consists of six consecutive dominant bits. Since this "violates" bit stuffing rules, it is used as an error indicator to the system (see coding/decoding).

An Error Flag is transmitted if an 82526 operates as an error active node and has detected an error condition during or after a message transfer. If an Error Flag is generated by a transmitter, or a receiver, all other nodes interpret the Error Flag as a bit stuffing rule violation. As a consequence, they, in turn, transmit an error flag. A variable sequence of dominant bits result from the superposition of the different Error Flags transmitted by individual nodes. The total length of the Error Flag sequence varies between six bits minimum to twelve bits maximum.

An error condition is signaled by the transmission of six recessive bits while in the error passive operation mode. This way an error passive node with a temporary local receiver problem will not destroy messages received correctly by other nodes. The recessive bits may be overwritten by an Error Flag generated by one or more error active system nodes, but the error passive 82526 waits for at least six bits of equal polarity before entering into the next internal receive or transmit mode. (See Error Handling for error active/passive mode.)

NOTE:
The 82526 will not perform storage of a message (positive acceptance filtering) into the communication buffer, if reception of the message was followed by an Error Flag on the serial bus.

The error-delimiter consists of seven recessive bits generated by the 82526 after the end of an Error Flag on the serial bus line. This is monitored by detection of a transition from the dominant to recessive bit level.

Detected errors during the transmission of a data or remote frame can be signaled within the transmission time of the respective frame. This procedure associates an Error Flag to the corresponding frame, and initiates a retransmission of the frame. As the 82526 monitors, any deviation of its error frame will start retransmitting an error frame. If this occurs several times in a sequence the 82526 will become error passive.

4.4 Overload Frame

The overload frame consists of two bit fields, the overload flag and the overload delimiter.

There are two cases of overload conditions which result in the transmission of an overload flag:

1. Internal conditions of the receiver circuitry of the 82526 which require a delay time before receiving the next frame (receiver not ready).
2. Detection of a "dominant" bit during Interframe Space.

The overload frame consists of six dominant bits that correspond to the Error Flag and destroy the fixed form of the Interframe Space Field. As a consequence, all other nodes see the dominant bit during the Interframe Space time and interpret the recessive to dominant edge as a start of frame and transmit an overload flag because of the overload condition.

The overload delimiter consists of seven recessive bits generated by the 82526.

After transmission of an overload frame, each 82526 within the system monitors the bus line until a transition from a dominant to a recessive level occurs. This indicates to each 82526 the end of overload frames and each node simultaneously starts the transmission of six more recessive bits.

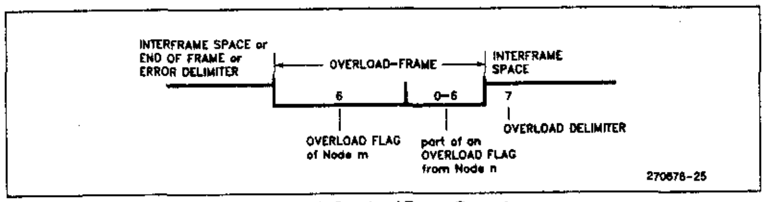


Figure 12. Overload Frame Format

NOTE:
The earliest time an overload frame can be transmitted is at the first bit time of the Interframe Space Field. This is contrary to the Error Frame and allows the 82526 to differentiate between the Error Frame and Overload Frame.

Interframe-Space

Data Frame and Remote Frame are separated from preceding frames by an Interframe Space consisting of the Intermission bit field and a possible Bus Idle time. An error frame is not preceded by an Interframe Space.

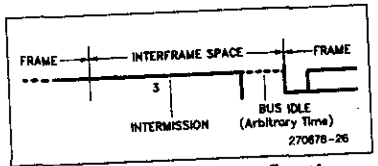


Figure 13. Interframe Space Format

INTERMISSION consists of three recessive bits. During Intermission time the 82526 will not start transmission of a frame. Intermission is a fixed time period for the 82526 to execute internal processes prior to the next receive or transmit task.

Data received within a data frame will be stored in the communication buffer and the control bits are updated if no error condition has occurred through the last bit of the end of frame field.

The bus idle time may be of arbitrary length. After the Interframe Space period, the 82526 looks for bus idle before initiating transmission, if requested by a CPU. The detection of a dominant bit after Intermission or bus idle is interpreted by the 82526 as Start of Frame.

5.0 CODING/DECODING

5.1 Coding

The frame segments (start of frame, arbitration field, control field, data field and CRC sequence) are coded using bit stuffing. Whenever the transmit logic of the 82526 detects five consecutive bits of identical levels to be transmitted, the logic inserts a complement bit in the transmitted bit stream.

Bit stuffing is used to guarantee enough edges in the NRZ Bit Stream to maintain synchronization.

5.2 Decoding

Whenever the 82526 has received five identical consecutive bit levels in the received bit stream the logic automatically deletes the next bit from the data stream (destuffing). Some field formats do not use bit stuffing. In these cases the bit stuffing and destuffing logic is turned off.

6.0 ARBITRATION

In the case when two or more 82526s start transmission concurrently, the bus access conflict is solved by a bit-wise arbitration method during transmission of the arbitration field.

The transmit logic compares the bit level transmitted to the level monitored on the serial bus. Both the transmitter and receiver are on the bus at the same time. The transmit logic stops message transfer if a recessive bit was sent but a dominant bit was monitored. This method guarantees transmission of the message with the highest priority even if there is a collision during the arbitration field of one or more message Identifier(s).

The 82526 protocol architecture requires each message used in the communication network to have a unique Identifier characterizing the type of data within the data field. Using this method, the Identifier assigns a name to the data frame and automatically implies the priority of the message.

As a result, the Identifier during bus access represents not only the message name but, more important, the priority of each specific message. Since the most significant bit (MSB) of an Identifier is transmitted first, the Identifier with the smallest digital value has the highest priority for bus access.

An Identifier should not be used for more than one specific message (total available # of Identifiers = 2032) to ensure that two or more nodes never simultaneously start a transmission of a data frame with the same message priority. Following this rule, bus access conflicts are resolved during the transmission of the Identifier.

One exception would be the simultaneous transmitter and receiver initiated frame transfer for the same message. If one 82526 generates a request for actual data of a certain type by transmitting a remote frame and simultaneously, the 82526 responsible for this type of data starts the transmission, arbitration can not be solved by the Identifier itself and actually is not required.

To deal with this exception, the RTR-bit is included in the arbitration field. The RTR-bit of the transmitter is always set dominant and, therefore, has a higher priori-

ty than the requesting 82526 (RTR-bit recessive). This way the remote frame request by the receiver gets an immediate response by the transmitter.

Example: Non-Destructive prioritized bitwise arbitration

START BIT
MESSAGE IDENTIFIER FIRST BIT

NODE 1:	0	0	0	1	0	1	1	0	0	0	1	ENGINE SPEED	(081H)
NODE 2:	0	1*	1	0	0	1	0	1	1	0	1	COOLANT TEMP	(32DH)
NODE 3:	0	0	0	1	1*	0	1	1	0	1	0	VEHICLE SPEED	(00AH)
BUS:	0	0	0	1	0	1	1	0	0	0	1		(081H)

270678-27

NOTES:
Node 1 wins the arbitration with the lowest identifier (081H)
• Zero is the dominant bit on the bus
*Node detected the loss of the arbitration and stopped transmitting but continues to receive the message.

The CAN protocol architecture defines that each Communication Object used must have a unique Identifier characterizing the priority of the message. This allows bitwise arbitration of the bus if a conflict arises. The transmit logic compares the level monitored on the serial bus with that transmitted. The transmit logic immediately stops transmission if there is a conflict. This guarantees the data transfer of the Communication Object with the highest priority even if there is a collision.

7.0 ERROR DETECTION AND FAULT CONFINEMENT

The Error Detection mechanism is implemented in hardware for efficiency.

7.1 Bit Error

During a transmit operation, the 82526 monitors the bus on a bit-by-bit basis. If the bit level monitored is different from the transmitted bit, a bit error is signaled.

Exceptions: Arbitration and ACK-SLOT. During arbitration, a recessive bit can be overwritten by a dominant bit. In this case, the 82526 interprets a bit error as an arbitration loss. During the ACK-SLOT, a transmitter may detect a falsified bit (recessive to dominant) meaning that at least one receiver has received the message correctly.

NOTE:
Except during transmission of the arbitration field and during the time window of the ACK-SLOT, all global and local errors at the transmitter are detected.

7.2 Bit Stuffing Error

As described earlier, the frame segments are coded by a method of bit stuffing.

There are two possibilities where bit stuffing errors may occur:

1. A disturbance generates more consecutive bits of equal level than allowed by the rule of bit stuffing. These errors are detected by all nodes.
2. A disturbance falsifies one or more of the five bits preceeding the stuff bit. This error is not recognized by a receiver; however, if the error also appears at the transmitter, it will be detected as a bit error (transmitter monitors bus as it transmits).

In any case, the error is detected by a receiver either by the bit stuffing mechanism (the stuff bit of the transmitter is not dropped but taken as an information bit) or by the CRC check.

7.3 CRC Error

To ensure the validity of a transmitted message, all receivers perform a CRC check. In addition to the information bits, the CRC includes control bits used for error detection.

Description of the CRC Code

The code used for the 82526 is a (shortened) BCH Code, extended by a parity check and the following attributes:

— 127 bits as maximum length of the code word
— length of the CRC sequence is 15 bits
— Hamming distance d = 6
  d = min A (x EXOR y) / x, y different code words
  A(x) = number of "recessive" bits in the code word x

The CRC SEQUENCE is determined by the request that the code word, if interpreted as polynomial with coefficients 0 or 1 is divisable by the polynomial.

$$f(x) = (x^{14} + x^{9} + x^{8} + x^{6} + x^{5} + x^{4} + x^{2} + x + 1)(x + 1)$$

$$f(x) = 1100\ 0101\ 1001\ 1001$$

$$f(x) = \text{C599 Hex}$$

Burst errors are detected up to a length of 15 (degree of f(x)). Multiple errors (number of disturbed bits at least d = 6) are not detected with a residual error probability of 3×10^{-5}.

7.4 Form Error

Form Errors result from the violation of the fixed form of the following bit fields:

— end of frame
— interframe space
— ACK delimiter
— CRC delimiter

During the transmission of these bit fields, an error condition is recognized if a "dominant" bit level is detected.

7.5 Error Detection Capabilities

Global errors, which occur at all fully functional nodes, are 100% detectable.

For local errors, e.g. errors which may appear at some nodes only, the shortened BCH Code extended by the parity check has the following error detection capabilities:

— Up to 5 single bit errors are detected 100% even if those errors are being distributed randomly within the code word.
— All single bit errors are detected if their total number within the code word is odd.
— The residual error probability of the CRC check is $2^{-15} = 3 \times 10^{-5}$. As an error may be detected by the CRC check, and/or by additional implemented error detection mechanism, the residual error probability is significantly less than 3×10^{-5}.

7.6 Fault Confinement

Fault Confinement is implemented on the 82526 as a self-checking mechanism for distinguishing "temporary errors" from "permanent failures". A permanent failure is noted when an average of one in eight messages is corrupted. This type of error condition can be caused by a defective connector, transmitter, receiver or a long lasting disturbance from outside the network. If the node continues to observe a failure over a period of time the node will be removed from the bus. A disconnected node is not placed again on the serial bus until the CPU has issued a software reset to the 82526, and an 82526 internal delay time has elapsed.

The implementation of the fault confinement consists of two counters (RECEIVE-ERROR-COUNT and TRANSMIT-ERROR-COUNT) and some control logic. These counters are modified according to a number of rules, which may be considered as the core of the error confinement. If a message is transmitted or received without an error the error-counter is decremented by a fixed number, if it is not already 0. The error-counter is increased by a fixed number, if an error is detected on the serial bus. No access is provided to the ERROR-COUNTERS; however, two flags are provided (Error Status and Bus Status) as a summary of error events.

The count added to the error-counter depends on the type of the error detected. For instance, whenever a node detects and reports an error condition (error flag), all system nodes will also detect an error condition due to that error flag, even if the information up to that time was received error-free.

NOTE:
In case an error condition is not detected by all nodes at the exact same bit time, the node reporting the error first is more likely to be responsible for such an error condition compared to those nodes reacting to the error flag. Therefore, a node that is often responsible for error conditions, as mentioned above, is the origin of the error as a result of a defect.

In general, a defective node exchanges information for a short time so as to prevent it from loading the bus and slowing down other nodes on the bus.

Three error states are flagged in the STATUS-REGISTER as follows:

ERROR-STATUS

Error status can be set because of disturbances of either the receive path or transmit path but this information is not given to the user.

If ERROR-STATUS is set, the bus is recognized as being disturbed. Whenever the ERROR-STATUS bit changes and the ERROR-INTERRUPT is enabled an ERROR-INTERRUPT is issued to the CPU.

DPRAM-STATUS

If DPRAM-STATUS is set, the DPRAM setup is recognized as being incorrect. The reason may be a missing ENDMARK, mismatch of data and length, or an invalid concatenation of the communication objects.

Whenever DPRAM-STATUS is set, BUS-STATUS is also set. The node is turned off from the bus and an interrupt is generated if ERROR-INTERRUPT is enabled.

BUS-STATUS

BUS-STATUS indicates to the CPU that the node is in the BUS-OFF state and must be reset to be placed back on the bus.

An 82526 in the BUS-OFF state will neither transmit nor receive messages. In order to restart the 82526 it is necessary to:

1. Start the RESET sequence, beginning with RESET-REQUEST set. Check that the DPRAM setup is correct
2. Restart the 82526 by removing the RESET-REQUEST

NOTE:

After this sequence, the CAN node will be active on the serial bus after:

— 11 consecutive recessive bits on the serial bus in case DPRAM-STATUS was set

— 128 * 11 consecutive recessive bits in case BUS-STATUS was set but DPRAM-STATUS was not. These 128 × 11 recessive bits represent the internal BUS-OFF delay time

7.7 82526 States with Respect to the Serial Bus

The 82526 may be in one of the three following states:

— Error Active

— Error Passive

— Bus Off

Error-Active is the normal mode of operation, it is characterized by the 82526 transmitting an ERROR-FLAG of 6 dominant bits in case a receive or a transmit error is detected.

An "Error Passive" 82526 does not send an ACTIVE ERROR FLAG (6 dominant bits). In this mode, the 82526 communicates on the bus but on detecting a receive or a transmit error, it sends a PASSIVE ERROR FRAME (6 recessive bits). After transmitting a message, an "Error Passive" 82526 does not initiate another transmission immediately; instead, after the INTER-FRAME SPACE, it transmits 7 "recessive bits". If during this time period (7 recessive bits), another 82526 starts transmission on the bus, the "error passive" 82526 becomes the receiver. The "error passive" state is indicated to the user as a pre-warning by the Error-Status bit in the Status Register (Add. 01H).

A Bus-Off 82526 does not transmit or receive any information; its output drivers are put in a float state. This state is indicated to the user by the BUS-STATUS bit in the Status Register (Add. 01H).

For Fault Confinement two error counters are implemented in the 82526 architecture:

1. TRANSMIT AND RECEIVE ERROR COUNTER

These counters are modified by the 82526 with no read or write access to the user.

At the system start-up, there may be only one 82526 active on the bus. If this node transmits a message, it will not get an acknowledgement and hence detect an error; it will correspondingly repeat the message. Because of repeated transmissions, such 82526 will become "Error Passive" but not "Bus-Off".

Similarly, an 82526 sending a wake-up message may become "Error-Passive."

8.0 PORT REGISTERS AND CLOCK DIVIDER REGISTER

8.1 Port Data Register

Port 0 and Port 1 of the 82526 are implemented in a way that allows the user to define the inputs and outputs of a port within the data itself. A written "1" to a port data register either means the appropriate pin is set high as an output function or this pin is configured as input. If a pin has been written to a "1", it may be read correctly.

Address	MSB							LSB
FEH	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
FFH	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0

8.2 Clock Divider and $\overline{CS}$ 0–2 Programming Register

The 82526 provides a clock output (CLKOUT) to drive a CPU or external logic in the external clocked mode. CLKOUT is derived from the internal 82526 clock oscillator by a programmable divider register shown below:

Address	MSB							LSB
FDH	CSPROG	X	X	X	CCDR3	CCDR2	CCDR1	CCDR0

After the reset, the default value of this register is 0xxx0000B.

8.2.1 CLOCK DIVIDER

CCDR

3210	freq (CLKOUT)
0000	XTAL/2 Default Value after Reset
0001	XTAL/4
0010	XTAL/6
0011	XTAL/8
0100	XTAL/10
0101	XTAL/12
0110	XTAL/14
0111	XTAL/16
1000	XTAL/18
1001	XTAL/20
1010	XTAL/22
1011	XTAL/24
1100	XTAL/26
1101	XTAL/28
1110	XTAL/30
1111	XTAL (not XTAL/32)

8.2.2 $\overline{CS}$ 0–2 PROGRAMMING

The outputs are push-pull and are programmable by CSPROG.

CSPROG = 0

The $\overline{CS}$ outputs become valid after the address is latched (on the falling edge of ALE), if address is in the range of these outputs. This is the default state after the reset.

CSPROG = 1

The CS outputs become stable and valid in response to the address on the AD0–AD7 after the ALE goes high.

For detailed timings, please refer to the diagrams in the **82526 Data Sheet.**

9.0 SET UP FLOW EXAMPLE

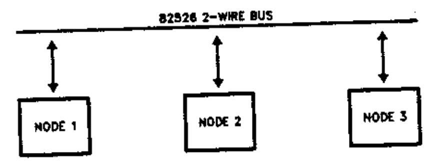
The following example allows Node 1 to receive data from Node 3 and Node 2 to transmit data to Node 3. Setups for each of the three Nodes follow.

This Diagram shows the data flow for the following example. Node 3 receives and transmits data, node 1 only receives and node 2 only transmits.

The Communication Object ID 4 is used to transmit data from node 3 to node 1. A Communication Object ID of 5 is used to transmit data from node 2 to node 3.

270678-28

This Diagram shows the physical layout of the three nodes.



270678-29

NODE 1 SETUP: (Receive one byte of data from Node 3)

6

NODE 1 SETUP (Receive one byte of data from Node 3)

NOTE:
The IMP must be in Reset before the register setup starts

DPRAM Address	Description	Data	Comments
00H	CONTROL REGISTER	0FH	Reset Request
01H	STATUS REGISTER	00H	
02H	INTERRUPT POINTER	FFH	Initialize
03H	BUS TIMING REGISTER 0	00H	BRP = 0, SJW = 0
04H	BUS TIMING REGISTER 1	12H	
05H	OUTPUT CONTROL REGISTER	AAH	Normal Mode Data on Both of 2-Wire Bus
06H	DESCRIPTOR BYTE 1 OBJECT 1	80H	Message ID = 4
07H	DESCRIPTOR BYTE 2 OBJECT 1	40H	
08H	DESCRIPTOR BYTE 3 OBJECT 1	21H	Only Reception of Message. Interruption Receive 1 Data Byte.
09H	DATA BYTE 1		
0AH	END MARK (FFH)		

After this setup, the CPU must reset the Reset Request Bit in the Control Register (Add. 0H) to 0.

NODE 2 SETUP (Transmit one byte of data to Node 3)

NOTE:
The IMP must be in Reset before the register setup starts

DPRAM Address	Description	Data	Comments
00H	CONTROL REGISTER	0FH	Reset Request
01H	STATUS REGISTER	00H	
02H	INTERRUPT POINTER	FFH	Initialize
03H	BUS TIMING REGISTER 0	00H	BRP = 0, SJW = 0
04H	BUS TIMING REGISTER 1	12H	
05H	OUTPUT CONTROL REGISTER	AAH	Normal Mode Data on Both of 2-Wire Bus
06H	DESCRIPTOR BYTE 1 OBJECT 1	80H	
07H	DESCRIPTOR BYTE 2 OBJECT 1	54H	ID = 5 Transmission of Data Frame
08H	DESCRIPTOR BYTE 3 OBJECT 1	A1	One Data Byte
09H	DATA BYTE 1		
0AH	END MARK (FFH)		

After this setup, the CPU must reset the Reset Request Bit in the Control Register (Add. 0H) to 0.

The following 82526 setup will transmit one byte of information to the receive object above (Node 1) and receive one byte of information from the transmit object above (Node 2).

NODE 3 SETUP (Transmit to Node 1 and Receive from Node 2)

NOTE:
The IMP must be in Reset before the register setup starts

DPRAM Address	Description	Data	Comments
00H	CONTROL REGISTER	0FH	Reset Request
01H	STATUS REGISTER	00H	
02H	INTERRUPT POINTER	FFH	Initialize
03H	BUS TIMING REGISTER 0	00H	BRP = 0, SJW = 0
04H	BUS TIMING REGISTER 1	12H	
05H	OUTPUT CONTROL REGISTER	AAH	Normal Mode Data on Both of 2-Wire Bus
06H	DESCRIPTOR BYTE 1 OBJECT 1	80H	
07H	DESCRIPTOR BYTE 2 OBJECT 1	50H	Rec of ID = 5
08H	DESCRIPTOR BYTE 3 OBJECT 1	21H	
09H	DATA BYTE 1	XXH	Data
0AH	DESCRIPTOR BYTE 1 OBJECT 2	80H	
0BH	DESCRIPTOR BYTE 2 OBJECT 2	44H	Transmit ID = 4
0CH	DESCRIPTOR BYTE 3 OBJECT 2	A1H	
0DH	DATA BYTE 1	XXH	Data
0EH	END MARK (FFH)		End Mark

After this setup the CPU must write the Reset Request bit in the Control Register (Add. 0H).

10.0 FLOW DIAGRAMS

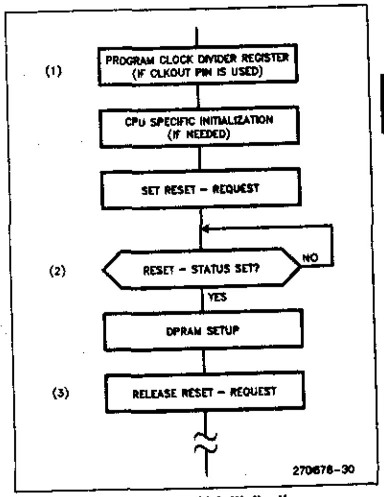
10.1 Flow 1

82526 INITIALIZATION

1. Process clock-out from default value of XTAL/2 to desired operating frequency if CLKOUT pin is used.
2. A Reset-Request does not force the 82526 to enter into the reset state immediately if already in operating mode. A transmission or reception in progress is finished first. Therefore, the CPU has to wait for RESET-STATUS to be set by the IMP as acknowledgement of the RESET-REQUEST, in the Status Register.

NOTE:
For the production part of the 82526 use HALT-REQUEST rather than RESET-REQUEST to stop the IMP and to change the buffer memory configuration. RESET-REQUEST must be used for any start-up routine but in operating mode would result in a reset of the EML logic that is not recommended.

3. Start 82526 operation.

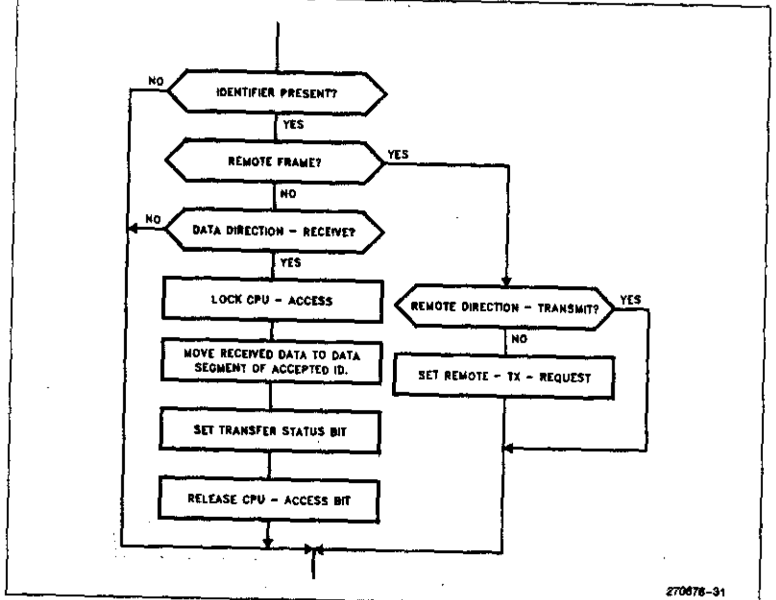


Flow 1: For 82526 Initialization

6

10.2 Flow 2

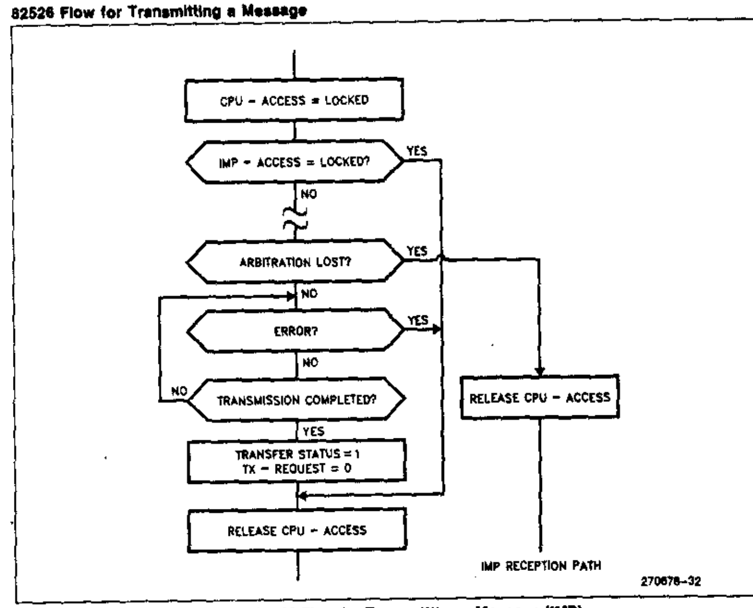
82526 Flow for Receiving a Message



Flow 2: 82526 Flow for Receiving a Message (IMP)

10.3 Flow 3

82526 Flow for Transmitting a Message

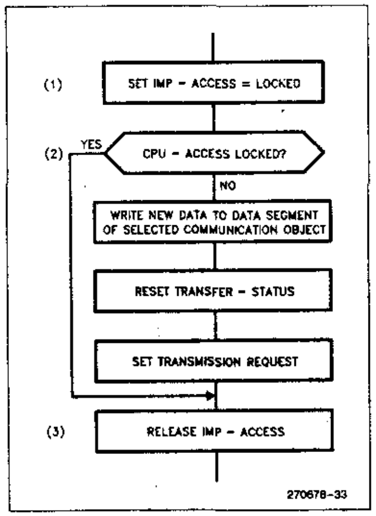


Flow 3: 82526 Flow for Transmitting a Message (IMP)

10.4 Flow 4

CPU FLOW FOR TRANSMISSION OF A MESSAGE

1. The transmission of an object must not start while updating data or control bits.
2. Active wait for CPU-ACCESS released as in the reception flow is not practical as CPU-ACCESS might be locked for as long as 135 bit times.
3. IMP-ACCESS must be released. If IMP-ACCESS is locked for a communication object with TRANSMISSION-REQUEST set, this and any communication object with a higher descriptor address will not be transmitted.

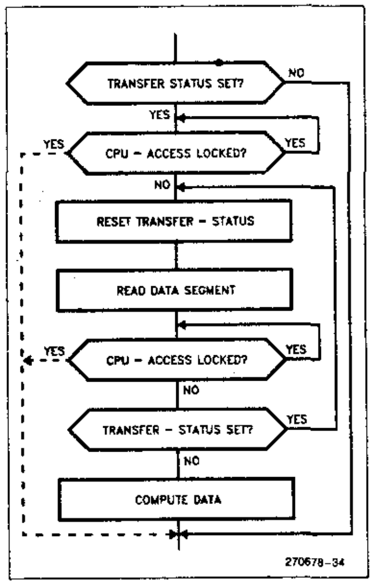


Flow 4: CPU Flow for Transmission of a Message

10.5 Flow 5

CPU POLLING FLOW FOR RECEPTION OF A MESSAGE

1. If CPU-ACCESS is locked, the IMP currently updates either data or control bits. The time CPU-ACCESS remains locked might last up to 100 crystal 82526 XTAL1 - cycles.
2. CPU-ACCESS being locked after data has been read means that the IMP is updating this specific data segment. As shown in Flow 2, the IMP does not check the IMP-ACCESS bit but moves data to the appropriate data segment when reception is validated.
3. New data is moved to this specific data segment by the IMP during multiple read accesses (2 or more) by the CPU. As a result, read data might be inconsistent and therefore need to be read again.
4. If active wait is not applicable.
5. In case there is only a one byte data segment, steps 2), 3), and 6) are not required.
6. Necessary to guarantee data consistency for multiple byte data segment.

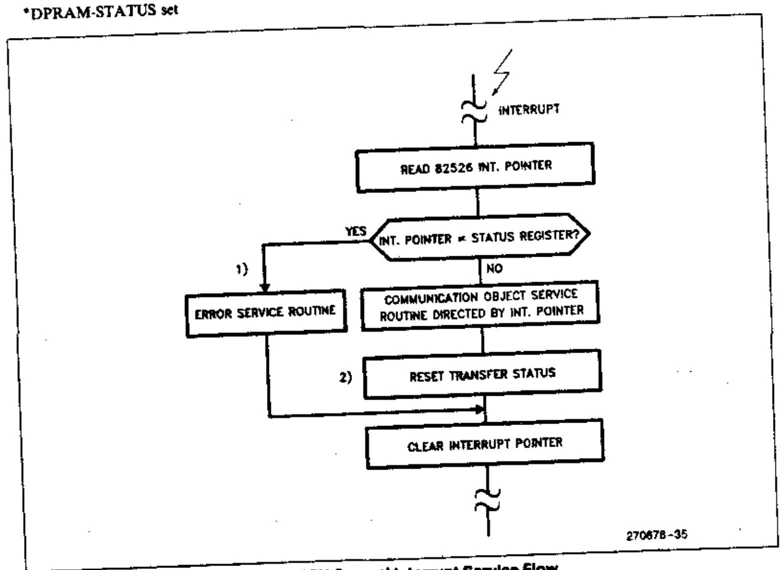


Flow 5: CPU Polling Flow for Reception of a Message

10.6 Flow 6

CPU GENERAL INTERRUPT SERVICE FLOW

1. There are several possible cases for error interrupts:
   - *ERROR-STATUS bit modified (set or reset)
   - *BUS-STATUS = off-bus
   - *DPRAM-STATUS set
2. Reset of the TRANSFER-STATUS bit normally is included in the communication object service routine. It is of major importance to reset this bit before clearing the interrupt pointer. Therefore, this action can be found in the flow explicitly. If the interrupt pointer is cleared prior to TRANSFER-STATUS reset, the CPU can be interrupted again for the same transfer.



Flow 6: CPU General Interrupt Service Flow

6

82526
CONTROLLER AREA NETWORK CHIP
Automotive

- **On-Board "MOTEL"**
- **Multimaster Architecture**
- **Bus Access Priority by Message**
- **2032 Different Message Objects**
- **Guaranteed Latency Time for High Priority Messages**
- **Powerful Error Handling**
- **Data Length to 8 Bytes**
- **Message Configuration Flexibility**
- **Broadcast Message Transfer**
- **Ready Output**
- **Global Interrupt Disable**
- **Non-Destructive Bitwise Arbitration**
- **Two 8-Bit I/O Ports**
- **NRZ Coding/Decoding with Bit Stuffing**
- **Programmable Transfer Rate to 1 MBit/Sec**
- **Programmable Output Driver Configuration**
- **Programmable Clock Output**
- **44-Pin PLCC**
- **Three Additional $\overline{CS}$ Outputs**

The 82526 Communication Controller is a highly integrated VLSI device which implements the CAN (Controller Area Network) Protocol. Included on the chip are an Interface Management Processor (IMP), Bit Stream Processor (BSP), Bus Timing Logic (BTL), Transceiver Control Logic (TCL), Processor Interface Unit (PIU), Error Management Logic (EML), Clock Generator and a pseudo Dual Port RAM (DPRAM). These hardware modules implement all necessary features of a high performance serial communication protocol. When connected to a microprocessor, the 82526 performs the principal functions of the physical and data link layer. Figure 1 shows a block diagram of the 82526.

The 82526 uses an 8-bit multiplexed address and data bus optimized for operating with Intel's microcontrollers and microprocessors. Other architectures may also be used with the direct connect on board "MOTEL" circuitry.

As shown in Figure 1, the BSP, TCL, BTL and EML are related with Bus Line Logic. The IMP, RAM and PIU are related to the CPU Interface Logic. The logic blocks BSP, BTL, TCL and EML are referred to as the "Serial Interface Unit".

The CPU communicates with the 82526 through the on-chip pseudo dual port RAM which includes global status and control registers. The on-chip memory serves as the communication buffer interface between the CPU and the IMP. The CPU initializes the global status and control registers and creates a data structure (Communication Objects) within the communication buffer for reception and transmission of defined messages.

Commands, data and status transfers take place over the 8-bit parallel bus. The CPU writes data to the 82526 using $\overline{CS}$, ALE and $\overline{WR}$ signals and reads the 82526 received data and status information using the $\overline{CS}$, ALE and $\overline{RD}$ signals. The 82526 uses the interrupt line to alert the CPU of errors or data received.

March 1991
Order Number: 270573-004

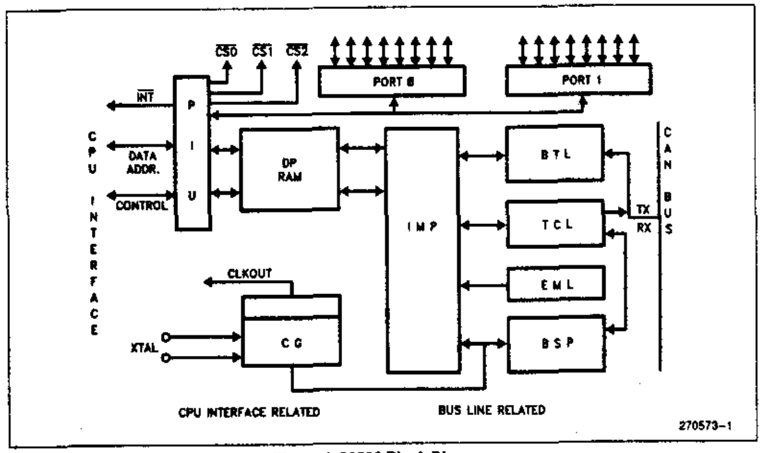


Figure 1. 82526 Block Diagram

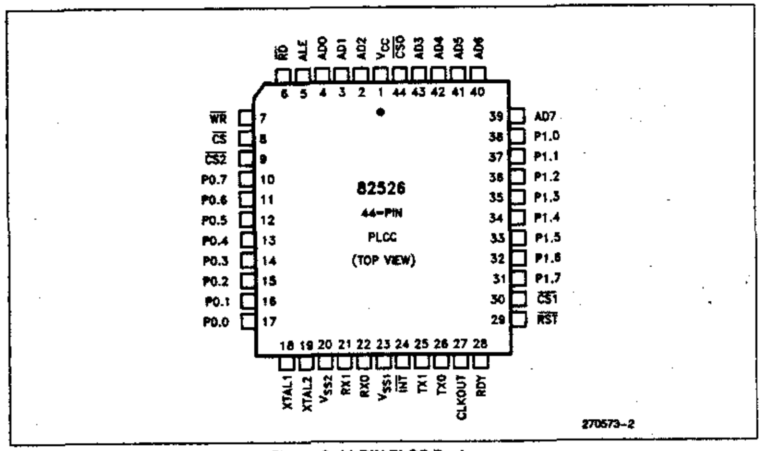


Figure 2. 44-PIN PLCC Package

ABSOLUTE MAXIMUM RATINGS

Ambient Temperature
under Bias .... −40°C to +125°C
Storage Temperature .... −65°C to +150°C

ELECTRICAL CHARACTERISTICS

D.C. CHARACTERISTICS V_{CC} = 4.5V to +5.5V; T_A = −40°C to +125°C

Symbol	Parameter	Min	Typ	Max	Conditions
V_{IL}	Input Low Voltage (All except XTAL1)	$V_{SS}-0.5V$		0.8V	
V_{IL1}	Input Low Voltage (XTAL1)	$V_{SS}-0.5V$		$0.2V_{CC}-0.1$	
V_{IH}	Input High Voltage (All except XTAL1, $\overline{RST}$, ALE, $\overline{CS}$)	2.0V		V_{CC} + 0.5	
V_{IH1}	Input High Voltage ($\overline{RST}$)	3.5V		V_{CC} + 0.5	
	Hysteresis on $\overline{RST}$	200 mV			
V_{IH2}	Input High Voltage (ALE, $\overline{CS}$)	0.5 V_{CC}		V_{CC} + 0.5	
V_{IH3}	Input High Voltage (XTAL1)	0.7 V_{CC}		V_{CC} + 0.5	
V_{OL}	Output Low Voltage (All Outputs except AD0–7, $\overline{INT}$, RDY, CLKOUT, TX0, TX1)			0.4V	I_{OL} = 1.6 mA
V_{OL1}	Output Low Voltage (AD0–7)			0.4	I_{OL} = 3.2 mA
V_{OL2}	Output Low Voltage (RDY, $\overline{INT}$)			0.4V	I_{OL} = 5 mA
V_{OH}	Output High Voltage (All Outputs except AD0–7, $\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$, TX0, TX1, CLKOUT)	2.5V			I_{OH} = −80 μA
V_{OH1}	Output High Voltage (AD0–7, $\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$)	0.6 V_{CC}			I_{OH} = −400 μA
		0.7V cc			I_{OH} = −80 μA
I_{LK}	Input Leakage Current (Except Port0/Port1)			±10 μA	$V_{SS} < V_{IN} < V_{CC}$
I_{IL}	Low Level Input Current (Port0/Port1)			−50 μA	V_{IN} = 0.4V
I_{TL}	Logical 1 to 0 Transition Current (Port0/Port1)			−750 μA	V_{IN} = 2V
I_{LT}	Latch-up Trigger Current			±80 mA	
C_{IO}	Pin Capacitance			10 pF	@1 MHz, 25°C
I_{CC}	Supply Current		22 mA	33 mA	fXTAL = 16 MHz
ISM	Sleep Mode Supply Current		1.2 mA 5.5 mA	2.2 mA 8.5 mA	fXTAL = 1 MHz fXTAL = 16 MHz

A.C. CHARACTERISTICS

Conditions: T_A = −40°C to +125°C, V_{CC} = 5V ±10%, V_{SS} = 0V, Port 2 outputs: CL = 150 pF, All other outputs: CL = 80 pF

Symbol	Parameter	Min	Typ	Max
1/tXTAL	Oscillator Frequency	1 MHz		16 MHz
tAVLL	Address Valid to ALE Low	20 ns		
tLLAX	Address Hold after ALE Low	12 ns		
tLLDV	ALE or $\overline{CS}$(1) Low to Valid Data Out			5tXTAL + 80 ns(8)
tRHDZ	Data Float after $\overline{RD}$ High	6 ns		50 ns
tRLDV	Valid Data Out Delay from Read Control			80 ns
tQVWH	Input Data Setup to $\overline{WR}$ High	20 ns		
tWHQX	Input Data Hold after $\overline{WR}$ High	18 ns		
tWHDV	$\overline{WR}$ High to Output Data Valid on Port 0 or Port 1			4tXTAL + 70 ns
tWHLL	$\overline{WR}$ High to Next ALE or $\overline{CS}$ Low(1, 2)	2tXTAL + 5 ns		
tWHWL	Time between Writes	4tXTAL + 5 ns		
tLLWH	ALE or $\overline{CS}$(1) Low to $\overline{WR}$ High	4tXTAL + 10 ns		
tALAL	Time between ALE Falling Edges (or $\overline{CS}$)(1)	8tXTAL + 5 ns		
tLLYV	End of ALE to RDY Setup			65 ns
tLLYH	End of ALE to RDY High(4) (with 1k External Pull-Up)	4tXTAL + 65 ns		6tXTAL + 65 ns
tLCSL	ALE or $\overline{CS}$ Low to $\overline{CS0}$-1-2 Low(1, 5)			45 ns
tHCSH	$\overline{CS0}$-1-2 High after ALE or $\overline{CS}$ High(3, 5)			70 ns
tAVCS	Address Valid to $\overline{CS0}$-1-2 Low(6)			40 ns
tcSHH	$\overline{CS}$ Active Hold after $\overline{RD}$ or $\overline{WR}$ High(7)	0 ns		

NOTES:
1. Whichever falling edge is last.
2. Some pipe-lining of the write accesses is possible. This spec is important mainly when processor speed is higher than 82526 speed (use of the RDY output . . .).
3. Whichever rising edge comes first.
4. RDY is an open drain output (so is $\overline{INT}$).
5. Timings valid on ALE falling or rising edges only if ALE is used to strobe the $\overline{CS0}$-1-2 outputs (default mode). If the other mode is programmed by writing 1 to bit 7 of address 253, the new state starts propagating to the $\overline{CS0}$-1-2 outputs as soon as a stable address has been decoded (providing that $\overline{CS}$ is also valid).
6. Timing valid only when ALE is not used to strobe the $\overline{CS0}$-1-2 outputs. In this mode, tHCSH also applies on $\overline{CS}$ rising edge and tLCSL on $\overline{CS}$ falling edge.
7. $\overline{CS}$ must be active low with $\overline{RD}$ or $\overline{WR}$ rising edge for proper completion of the read or write access.
8. If t_{LLDV} is too slow for the host microcontroller, a double read mechanism can be implemented. For details, contact an Intel representative.

6

Physical Layer Specifications Load Condition: 80 pF

RX0/RX1	Min	Max	Conditions
Input Voltage	V_{SS} − 0.5V	V_{CC} + 0.5V	
Common Mode Range	V_{SS} + 1.0V	V_{CC} − 1.0V	(1)
Differential Input Threshold	± 50 mV		(1)

TX0/TX1(5)	Min	Max	Conditions
Source Current	−3.0 mA		$V_{OUT} = V_{CC} - 0.4V$
	−6.0 mA		$V_{OUT} = V_{CC} - 1.0V$
Sink Current	10.0 mA		$V_{OUT} = 0.4V$
	20.0 mA		$V_{OUT} = 1.0V$
Maximum Permitted Source Current (TX0, TX1 Together)		−8.0 mA	
Maximum Permitted Sink Current (TX0, TX1 Together)		22.0 mA	
Rise Time		20 ns	(4)
Fall Time		15 ns	(4)

CLKOUT Specifications

CL = 50 pF	Min	Max	Conditions
CLKOUT frequency (1/T)	fXTAL/30(3)	16 MHz	
Rise Time(4)		15 ns	
Fall Time(4)		15 ns	
Output Low Voltage		0.2 V_{CC} − 0.1	
Output High Voltage	0.7 V_{CC}		
Output Duty Cycle is 50% if Clock Divider Register Not Equal	0FH		
Output Duty Cycle Equal XTAL If Clock Divider Register Equal	0FH		

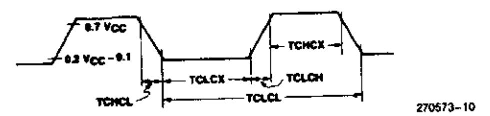
Internal Delay Time(2, 6)	Min	Max	Minimum Differential Input Threshold	Common Mode Range
$t_{(IN1)}$		167 ns	50 mV	V_{SS} + 1.0V to V_{CC} − 1.0V
$t_{(IN2)}$		137 ns	100 mV	V_{SS} + 1.0V to V_{CC} − 1.0V
$t_{(IN3)}$		127 ns	100 mV	V_{SS} + 1.5V to V_{CC} − 1.0V

NOTES:
1. See Internal Delay Times.
2. The Internal Delay Time is given by the sum: $t_{(internal\ logic)} + t_{(internal\ output\ driver)} + t_{(internal\ comparator)}$.
3. fOUT = fXTAL/R with R = 1,2,4,6,8, ... 2 × r and rmax = 15.
4. Measured between $0.2V_{CC} - 0.1$ and $0.7\ V_{CC}$.
5. Push-pull mode.
6. The total delay time from transmit to sample point = $t_{(total\ delay)} = t_{(internal\ delay)} + t_{(external\ logic)} + t_{(propagation\ delay)}$.

EXTERNAL CLOCK DRIVE

Symbol	Parameter	Min	Max	Units
1/TCLCL	Oscillator Frequency	1	16	MHz
TCHCX	High Time	20		ns
TCLCX	Low Time	20		ns
TCLCH	Rise Time		20	ns
TCHCL	Fall Time		20	ns

EXTERNAL CLOCK DRIVE WAVEFORM



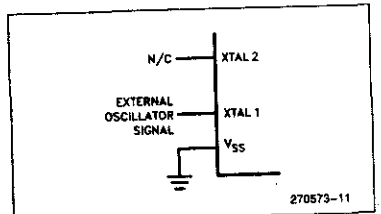
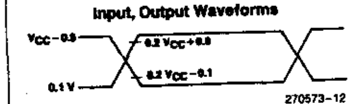


Figure 1. External Clock Drive Configuration

A.C. TESTING INPUT

Input, Output Waveforms



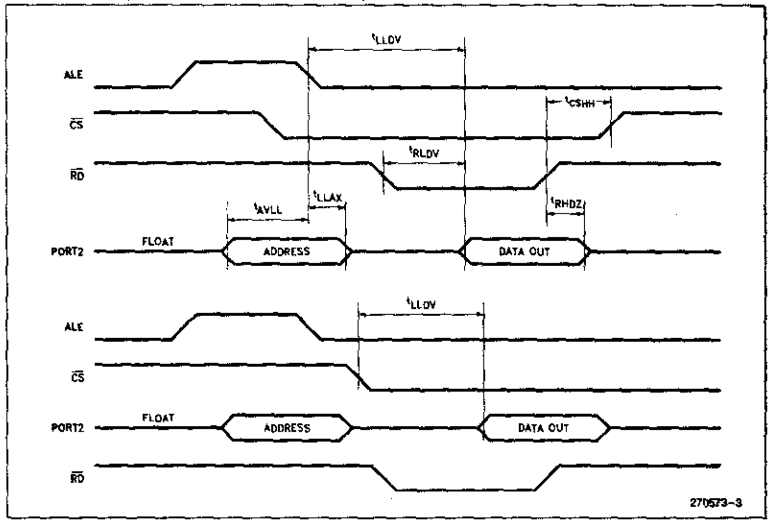
AC inputs during testing are driven at $V_{CC} - 0.5V$ for a Logic "1" and 0.1V for a Logic "0". Timing measurements are made at V_{IH} min for a Logic "1" and V_{OL} max for a Logic "0".

Float Waveforms

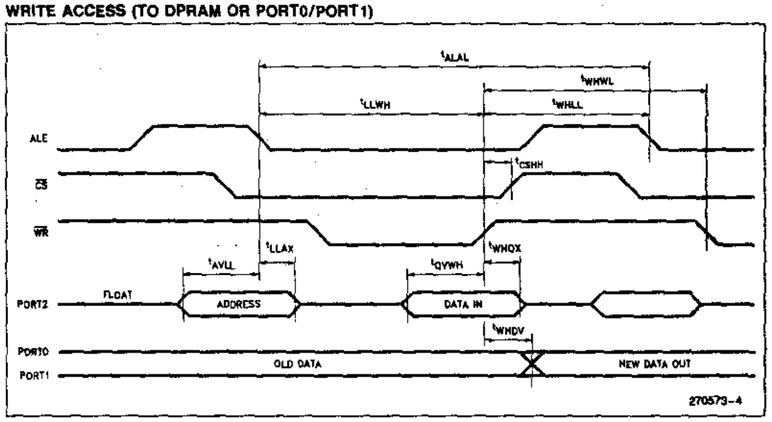
For timing purposes a port pin is no longer floating when a 100 mV change from load voltage occurs, and begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs. $I_{OL}/I_{OH} \geq \pm 20$ mA.

6

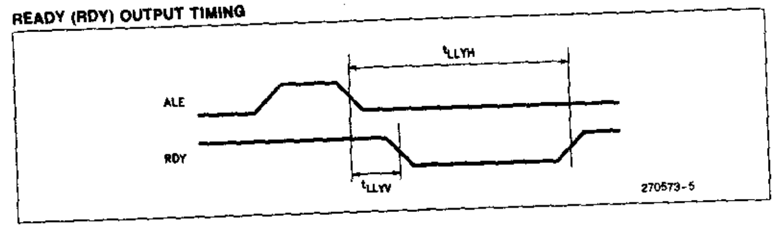
READ ACCESS (TO DPRAM OR PORT0/PORT1)



WRITE ACCESS (TO DPRAM OR PORT0/PORT1)

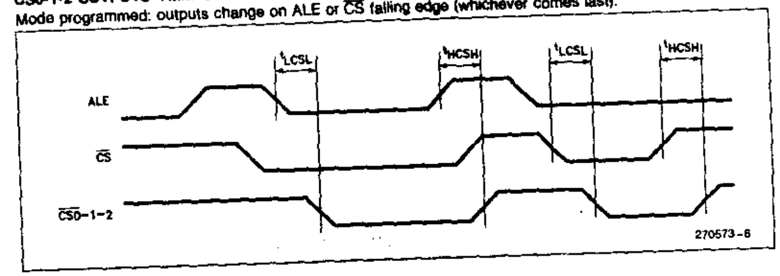


READY (RDY) OUTPUT TIMING



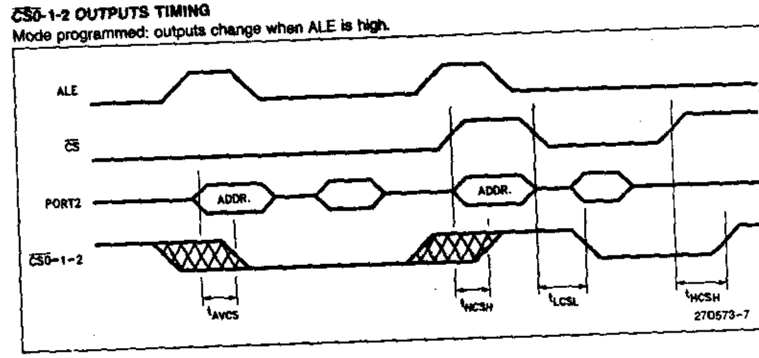
$\overline{CS0}$-1-2 OUTPUTS TIMING

Mode programmed: outputs change on ALE or $\overline{CS}$ falling edge (whichever comes last).



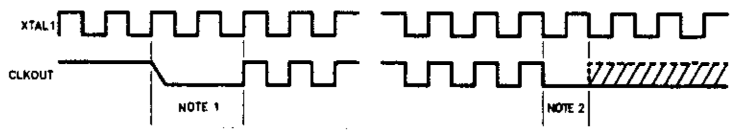
$\overline{CS0}$-1-2 OUTPUTS TIMING

Mode programmed: outputs change when ALE is high.



6

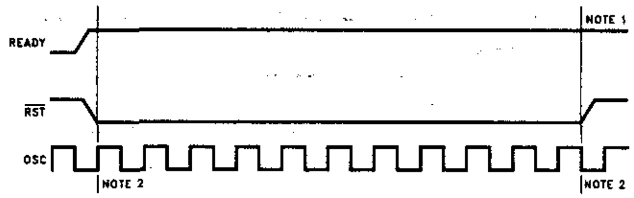
FREQUENCY TRANSITION TIME FOR CLOCKOUT LOGIC "0"



270573-8

NOTES:
1. $F_{OUT} = F_{XTAL}$/Divider Register $\rightarrow$ $F_{OUT} = F_{XTAL}$ max Logic "0" = 2 t_{XTAL}
2. $F_{OUT} = F_{XTAL}$ $\rightarrow$ $F_{OUT} = F_{XTAL}$/Divider Register
max Logic "0" = 1.5 t_{XTAL} If Divider Register = 2
max Logic "0" = (Divider Register − 1) XTAL If Divider Register > 2

RESET TIMING



270573-9

Minimum 10 Oscillator Cycles

NOTE:
1. Ready must not be pulled low during reset.
2. Reset timing measure from rising edge to rising edge of OSC.

MCS®-51 Architectural Overview

7